This is to certify that the

dissertation entitled

Design and Analysis of Local
Area Network Protocols For
Distributed Real-Time Systems

presented by

Taieb Znati

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Computer Science

[illegible]
Major professor

Date April, 1988

DESIGN AND ANALYSIS OF LOCAL AREA NETWORK PROTOCOLS FOR DISTRIBUTED REAL-TIME SYSTEMS

By

Taieb Znati

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

1988

ABSTRACT

Design and Analysis of Local Area Network Protocols for Distributed Real-Time Systems

By
Taieb Znati

Local area networks are steadily increasing their range of applications. While interconnection of computers and resource sharing has been typical of their applications, it is their envisioned role in distributed real-time systems which is receiving increased attention. The goal of this research is twofold: to design a multiaccess protocol for local area networks which better suits the requirement of a distributed real-time system; and to provide an analytical tool that can be used to study multiaccess protocols for bus networks in a unified manner.

Distributed real-time systems are characterized by their timing constraints. A substantial overhead is often incurred by the synchronization of the real-time processes running on different processors. This overhead can be reduced if there are guarantees on how fast messages are moved across the communication network. After examining the limitations of the standard protocols in providing such guarantees, we propose and study a virtual token protocol suitable for handling time-constrained messages in a distributed real-time system. The proposed protocol preserves the main advantages of the standard CSMA/CD and token passing standard schemes and remedies their principal shortcomings. As a result, the new protocol closely approximates an ideal priority scheme.

The global behavior of a class of multiaccess scheme for bus networks is characterized by a transmission phase, a resolution phase and an idle phase. We abstract a general mathematical model that can represent various multiaccess protocols for bus networks. The proposed model is based on the supplementary variables approach. We show that the use of this method results in a mathematically tractable system of state equations. While the proposed model is general, it can serve as a useful tool to compare various protocols for local area networks.

To my parents, *Fredj* and *Najia*, to my grandmother *Zenikha*,

and to the memories of my grandfather *Khemais*

ACKNOWLEDGEMENTS

I wish to express my gratitude to my thesis supervisor Dr. L. Ni for his continued support, enthusiasm and guidance during my Ph.D. program. His ideas and critical review of my research work were an incentive in the completion of this dissertation.

Dr. E. D. Goodman must be commended for his careful reading of this dissertation and for his insightful comments. His support and warm camaraderie, during my stay at the Case Center, are sincerely appreciated.

I would like to thank the other members of my committee, L. H. Greenberg, R. Lepage and S. Pramanik for their time and efforts.

I do wish to acknowledge members of my research group, Bruce M. McMillen, Eric Wu, Yew-Huey Liu, Chung-Ta King and Youran Lan, who have been there when needed.

Many friends, J. Carlson, C. Erickson, J. Foley, M. McPherson, S. Smith and M. Wilke, have made my stay at the Case Center very enjoyable. Special thanks go to A. V. Sannier for his proof-reading of parts of my thesis. I also would like to thank my friends S. R. Alani, F. Fathy J. S. Dhahir and C. A. Woodlock for their continuous support and encouragement.

Profound appreciation is extended to my parents, Fredj and Najia Znati, and to my brothers and sisters who believed in me. Their moral support was sometimes the only thing that kept me going.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the last decade, *distributed computing systems* have been rapidly gaining importance in the field of computing. The increasing interest in these systems is due to the recent advances in software, the rapid development of computer networking technology, and the availability of significantly reduced cost processing devices. The revolutionary changes in chip size-to-performance ratio brought by VLSI design techniques and the resulting improvement in processing storage and communications have induced penetration of the distributed computing systems into wider application domains [WiTi80]. The research reported in this thesis focuses mainly on the particular class of distributed systems which supports real-time applications. The goal of the research has been to design a communication protocol which fits better the requirements of these systems. A model to describe and analyze the performance of the real-time communication protocols in a unified manner is also developed.

In the remainder of this chapter, we review some issues related to distributed computing in general, regardless of the type of applications they support. We then briefly introduce the main characteristics of distributed real-time systems which are the focus of most of the discussion of later chapters. We next state the motivation of this research. We conclude the chapter by setting forth the plan of subsequent chapters.

1.1 AN OVERVIEW Of DISTRIBUTED COMPUTING SYSTEMS

The term distributed computing system applies to a wide range of system configurations, but usually refers to a collection of homogeneous or heterogeneous *computing capabilities* interconnected by a *communication subnet* and logically integrated in varying degrees by a *distributed operating system* [Efe78, Jens78, Stan84]. The physically distributed components of the system are inter-connected by a communications network. Depending on the geographical area covered by the distributed system, this network many be a long haul network or a local area network. Software sharing is facilitated through the use of communication protocol and information transfer. Distributed control implemented through a globally known set of policies integrates the physical and logical components of the distributed computing system into a functionally coherent system.

Two fundamental issues further characterize distributed systems: *system transparency* and *cooperative autonomy.*

System transparency provides the user with a logically integrated view of the system. The user should not be aware of the physical distribution of the system. To support this concept, the system must provide an interface which allows the description of a service without the need to specify the physical or logical component that provides the requested service. The user of the system should be able to request a service by name rather than by the designation of a server which provides that service.

Cooperative autonomy characterizes the degree of interaction among both physical and logical resources of the system. Every physical component and every peer process in the distributed architecture must have a fair degree of autonomy in managing its own resources. This cooperative autonomy is required in order to attain the objectives of the distributed system.

Distributed computing systems offer the potential for a significant number of benefits [Lee84]. These systems have the capability to provide increased system performance, improved resource sharing, improved reliability, graceful degradation and configuration flexibility.

Higher performance is achieved by permitting concurrent operations on the processors. The existence of multiple resources, both physical and logical, allows the system to avoid the bottlenecks that usually occur in a centralized system, thereby increasing the system throughput and reducing the response time.

Distributing control among the processors provides a practical means for sharing expensive resources, computing power, and information handling devices. The extension of communications among logical resources helps achieve a system-wide control of all activities, offering the potential for an optimal and dynamic resource allocation. In addition, system transparency hides the system status information so that the user does not need to know which resources are currently available and need not be concerned about how to request these resources in order to execute his tasks in the most performing way. This characteristic increases the system's capability to utilize its resources effectively.

System reliability and availability is achieved through the replication of resources and the reassignment of tasks from a failed processor to a fault-free processor. Furthermore, the distribution of control among multiple processors alleviates the system from the dependency on a single controller, allowing fail-soft service to a large class of users with different requirements and increasing the ability of the system to easily adapt to new configurations without significant disruption of the system.

Distributed computing systems, however, introduce new problems and raise new design issues. These problems are not only due to the system distribution but also to the heterogeneity of the computing elements and devices. The heterogeneous nature of distributed systems strongly affects several design issues.

As an illustration, the fact that each processing element has its own local names for different objects, introduces additional complexity to the design of the naming system. The difficulty arises from the need to create a global unique identifier space in a heterogeneous environment, where each local identification domain may use one or more different schemes of identifiers unique within itself. These problems are compounded by the difficulty to globally maintain an instantaneous distributed context and mapping information due to the delays and errors caused by the communication subsystems.

The reoccurring themes of heterogeneity, physical distribution, and multiple controlling authorities also significantly impact the protection needs in a distributed system. One of the major design goals of a distributed computing system is to provide its users with a uniform access to real and abstract objects. User interaction with these objects requires the implementation of a protection mechanism that ensures a controlled access to these resources. This protection must be based on rigid rules and concepts such as ownership, classification of levels of data security, or protection domains. The protection system must ensure that only authorized entities can read, modify, or manipulate resources. The system must also allow authorized entities to communicate and ensure that unauthorized entities cannot prevent authorized entities from manipulating the physical or logical resources of the system. Because of the need to interconnect different protection domains, each domain possibly implementing different protection policies, the task of designing a protection mechanism in a distributed system becomes more complicated. The implementation differences can be the source of mutual suspicion among systems. Therefore, the protection system must be built assuming a level of trust within the communication subsystem and within the single host and its physical and personnel security. The question remains to what extent the set of assumptions about what can be ultimately trusted could be considered safe and secure.

Distributed systems have come into existence in the computing field in some very natural ways. We certainly have not yet gained sufficient understanding of distributed systems. Our systems are still highly constrained and rigid in their construction and behavior. There are many hurdles to cross in designing a truly distributed system architecture. We need to develop a methodology for mechanizing the design and maintenance of distributed systems. This methodology must have provision for integrating fundamental issues such as access control, distributed control, reliability, heterogeneity, and efficiency. These issues have been recognized, but solutions to the problems they entail have not produced totally satisfactory systems.

1.2 DISTRIBUTED REAL-TIME SYSTEMS

As stated previously, distributed computing systems are proliferating rapidly and their widespread use has included several application areas. One of the most promising new areas for distributed systems is in the *real-time* systems domain. Distributed systems offer an attractive alternative for the implementation of real-time applications because of their potential for better performance, higher reliability and extensibility. In addition, for may real-time applications it is natural to distribute control among several connected processors. This characteristic facilitates the implementation of a distributed real-time application as a set of relatively independent processes which run asynchronously, except for occasional synchronization and communication [Moka85].

A real-time system is characterized by the timing requirements described in terms of deadlines by which processes must be completed. Although the completion time for a process is of importance in all types of systems, the response time in a real-time system is viewed as a crucial part of the correctness of the application software. In addition to implementing correctly the intended algorithms, real-time system must satisfy various timing constraints and respond in a timely fashion to external events or periodic update rates of the process variables under control. Usually, the computation involved for responding to external events is repetitive and may not be delayed beyond certain time limits.

Real-time systems have been divided into two classes: *hard real-time* systems and *soft real-time* systems [Moka83]. Hard real-time systems must continuously observe critical timing constraints. Failure to observe these constraints might bring about catastrophic results. Soft real-time systems can afford to occasionally miss their timing requirements causing eventually a degradation in performance but not a system failure.

1.3 MOTIVATION

A serious difficulty in meeting the requirements of distributed real-time applications is presented not only by the processor speed, but also by the efficiency of the communication network which interconnects these processors. As stated previously, the communication and synchronization of processes in a distributed real-time environment are achieved through the exchange of messages. A substantial number of messages may be required to maintain proper synchronization of the system processes. However, if the amount of communication overhead generated becomes excessive, the correctness of real-time processes, which depends on the adherence of these processes to the timing constraints imposed by the underlying application, may be jeopardized. Therefore, in order to allow an efficient scheduling of real-time processes, the network protocol must provide guarantees on how fast messages are moved across the communication network. This objective can be achieved by reducing the overhead incurred by the communication and by ensuring a predictable number of transition steps between message arrival and message departure. Consequently, any network protocol designed to support distributed real-time applications must be efficient and *deterministic*. Determinism is a crucial requirement in real-time environment. The guarantee of an upper bound on the access time is necessary in order to meet the timing constraints of different messages.

Establishing a finite upper bound for the message transmission delay solves only one part of the problem of how to meet timing constraints. An efficient scheduling of messages, most likely based on time dependent priorities or on explicit timing constraints such as deadline on the message transmission, will help achieve the real-time requirements. The priority will be used to associate costs to messages, so that the network message server can determine in which order to serve messages present in a waiting queue in an *optimal* fashion. We consider that an optimal scheduling is achieved when the number of lost messages is minimized.

Notice that in some applications, the loss of a message may cause the system to fail. In others, a small number of lost messages may be tolerable but will inflict a penalty upon the system such as disruptive effects, lack of certainty or other undesirable effects.

Determinism and the availability of an efficient priority scheme allow the communication system to guarantee adequate performance for various timing constraints. It provides the network with the capability to handle various traffic with various timing constraints, generated by different sources.

1.4 PROBLEM STATEMENT

The need for an efficient priority scheme in a real-time environment based on local area networks motivated the proposed research project. The study is aimed at developing a network access arbitration scheme for broadcast-type local area networks. Shared bus local area networks are attractive because of their relatively simple topology. Two medium access protocols for this type of networks for shared bus, CSMA/CD and control token, have received much attention. CSMA/CD based schemes provide a good throughput-delay performance at light load even for a very large number of stations. However, as the load on the channel increases, the probability of collision increases, leading to a poor channel throughput. In addition, the protocol lacks a guarantee for an upper bound on the transmission time of a message as well as a priority scheme; both requirements are crucial in handling real-time applications. Token-passing based protocols can achieve high channel throughput when the utilization is high. However, their performance decreases in a light load situation, due to the overhead generated by the token traveling among idle stations. From performance viewpoint, it is desirable to have a protocol that achieves good performance over the entire range of channel utilization. Moreover, in order to provide acceptable performance for real-time applications, a prioritized scheme is required. An ideal protocol for handling messages with different priority classes would combine the advantages of both CSMA/CD and token-passing bus protocols, but avoid their shortcomings.

The second objective of the research is to provide an analytical tool that can be used to study multiaccess protocols for bus networks in a unified manner.

1.5 DISSERTATION ORGANIZATION

In this Chapter, we argued the need for a communication network protocol that suits better the requirements of real-time applications. A better understanding of the fundamental issues and requirements of real-time systems must help the designer of real-time communication protocols identify the fundamental design requirements of these protocols. These issues and the problem relevant to the design of real-time access schemes are discussed in Chapter 2. We provide an overview and a classification of the major protocols proposed for local area networks. A better understanding of the capabilities and limits associated with these protocols in handling real-time applications is supplied. In Chapter 3 we propose an access arbitration scheme for real-time applications. The proposed scheme attempts to preserve the above mentioned advantages of CSMA/CD and token bus access protocols. In addition, the scheme intends to reduce the considerable portion of the token bus protocol devoted to necessary functions such as recovery from a token loss, due either to noise over the channel or to a station failure, a token duplication or resolving collision that results when more than one station tries to join the network at the same time. Above all, the scheme guarantees a more efficient way of handling different priority classes by implementing a priority mechanism that satisfies the basic requirements of a general priority scheme. Chapter 4 provides a formal description of the priority assessment phase. This phase is required to assess the priority of the access medium. Mechanisms related to the implementation of this phase are discussed. Dynamic as well as static approaches are proposed and their implementation issues are considered. Chapter 5 describes simulation experiments conducted to study the performance of the proposed protocol. The performance measures include the delay-throughput characteristics and the load versus throughput characteristics. Chapter 6 describes a unified analytical model to investigate the performance of multiaccess protocols. The application of the model to the proposed protocol is also provided. Chapter 7 contains the conclusion of the proposed work along with some remarks about avenues for future research.

Chapter 2

Design Requirements of Distributed Real-Time Systems

The purpose of this chapter is to identify the design requirements of distributed real-time systems. The basic issues involved in the design of these systems and the achievement of their objectives are discussed. The first part of this chapter is devoted to presenting some concepts which relate to the specifications and the design of distributed real-time systems. The fundamental problems introduced by these systems are identified and a discussion of their solution is provided.

The second part of the chapter is dedicated to discussing a specific part of these systems: the communication network. Issues related to the design of multi-access protocols for local area networks are examined in some detail. Real-time applications encompass different types of activities which emphasize different types of requirements. We attempt to identify the requirements that must be satisfied by any communication system intended for real-time applications. This analysis leads to a set of properties that must be satisfied in order to ensure proper functioning of the system. These properties include *robustness*, *flexibility*, *timing requirements* and *priority requirements*. A brief introduction of these issues and their relative importance in the design of local area networks for real-time applications is presented.

The last part of this chapter is dedicated to the particular problem of designing protocols for local area networks intended to support real-time applications. An overview of the currently available schemes is presented and their suitability for real-time applications is discussed.

2.1 DISTRIBUTED REAL-TIME SYSTEMS

The term real-time has been used to describe any information processing activity or system which has to respond to externally generated input within a finite and specifiable delay. Regulating a power plant, controlling a telescope, manipulating a robot arm, automatically controlling a flight in order to maintain the aerodynamic stability and reduce the peak of pressure on an aircraft are few of the many applications designed to perform in real-time. A major application in the field of computing is to control and monitor these systems.

It has been recognized that activities performed by real-time systems are usually loosely related and need not be done in a prescribed order [PoMi83, Moka85]. It becomes natural, therefore, to implement applications designed to monitor these systems as a set of relatively independent processes distributed among several processors. These processes can run asynchronously except for occasional synchronization and communication. The distributed nature of most real-time applications motivates the use of distributed architectures to handle real-time systems. However, real-time applications usually serve a limited geographic area. The extent of the area may still vary considerably from one application to an other, with substantial impact on the subcommunication system. Nevertheless, the timing requirements imposed by the application make the use of local area networks more suitable to handle distributed real-time systems than long haul networks. Local area networks are characterized by their high-bandwidth channels, which leads to shorter transmission delays, and their very low error rates. In addition, local area networks are more closely and easily controlled, as they are generally owned and operated by a single organization.

2.1.1 Specifications and Operating Characteristics of Real-time Systems

We view a distributed real-time system as being comprised of two parts: a controlled system and a controlling system. The controlled system consists of all the hardware which may be associated with laboratory analysis equipment, motors and mechanical devices, robotic arms, manufacturing assembly lines, or chemical processing units. The controlled system may be continuous or discrete in nature. The controlling system refers to the computing resources, hardware and software, that accept and analyze data from the controlled system. Specialized hardware is required to interface the controlling system with the sensors that measure process variables and with the actuators and other final control elements that manipulate the controlled system. Control algorithms which calculate changes to process variables depend strongly on the dynamic process characteristics such as feedback, feedforward, coordinated and sequence controls. These algorithms vary considerably in their complexity and their sensitivity to errors or variations in time. Data acquisition is typically carried out by many modules spread throughout the system. The acquired data is then recorded in a distributed database. Depending on the size of the system, the number of variables involved in the process may be quite extensive. Some of these process variables, such as temperatures, pressures, flow rates and levels are monitored on a *periodic* basis in order to control, display process status or perform process management and evaluation. Other processes, however, depend on digital signals, such as limit switches, status signals and relays that detect changes. These signals can generate random interrupts and so require *aperiodic* computations to be performed.

In order to accommodate both periodic and aperiodic requirements of general real-time applications, the controlling system must ensure certain basic capabilities. These capabilities allow distributed real-time applications to instantaneously record, analyze and respond to the raw data and events produced by the controlled system [Scho84]. More specifically, the controlling system must

- Support the creation, deletion and scheduling of multiple processes or independent tasks, each of which monitors or controls some portion of a total application,
- Provide communication primitives between processes to allow small amounts of data to be sent or received, thus allowing two or more related processes to share low-volume information,
- Allow data pooling or sharing, so that processes can pass and examine a large amount of data efficiently,
- Allow synchronization between processes in an application,
- Ensure the capability to quickly, reliably, and predictably maintain the data in long-term backing store,
- Allow synchronization with external events,
- Guarantee predictable and eventually instantaneous response to external events.

2.1.2 Timing Requirements of Real-time Computing

The attributes that characterize process variables are usually functions of physical timing. A set of internal laws, defined a priori, regulates the behavior of the physical process and constrains the process attributes to take their values in a specific domain. These *timing constraints* are usually described in terms of deadlines by which computations must be completed. The deadlines are generally traceable to the physical environment with which the system interacts and must be met to preserve specific properties of the physical process.

Some of the timing constraints emphasize the performance features of the system. Others relate more to its behavioral aspect. *Performance constraints* impose limits on the response time of the system. The capability to specify and enforce limits on the response time is crucial in the design of real-time applications and the achievement of the purpose for which they are designed. In particular, the response time may be sensitive to the instantaneous workload. The variability of response time in a real-time application may cause the system to miss its deadlines. In addition, variability in response time may result in poor performance of the system and user dissatisfaction. The guarantee that the system will invariably respond within a specified deadline substantially reduces the probability of error.

Behavioral constraints specify the rates at which stimuli are applied to the system. In the absence of the expected stimuli within the specified time, the system will take a specific course of actions. These requirements are used to invoke *artificial stimuli* which specify these actions. They describe the state the system must move into, or the response the system must produce in the absence of the expected stimuli.

Timing constraints can also be classified in terms of their rate and duration of occurrence. Dasarathy [Dasa85] identifies three such categories.

- Maximum inter-occurrence: this type of constraint specifies the maximum amount of time that may elapse between the occurrence of two consecutive events.
- Minimum inter-occurrence: this type of constraint specifies the minimum amount of time that may elapse between the occurrence of two consecutive events.
- Durational occurrence: this type of constraint specifies the amount of time units during which an event or a sequence of events must occur.

An event is defined to be either a stimulus to the system from its environment, or an externally observable response the system delivers to its environment. Notice that the above types are only applicable to some systems or to a particular set of events.

However, the three types are not necessarily exclusive. That is, a maximum and a minimum inter-occurrence as well as a durational occurrence can be simultaneously specified for an event or a set of events.

Timing constraints are a distinctive feature of distributed real-time systems. These requirements add new assertions to the correctness of real-time computations. These assertions involve the absolute timing of events. In other words, it is not enough to ensure that the computations involved in a real-time application are logically correct. In addition, the various timing constraints imposed by the application must be satisfied. Failure to adhere to these constraints results in a timing fault. The severity of the fault depends on the environment with which the system interacts. Hard real-time systems must respond in a timely fashion so as to meet stringent timing constraints and avoid catastrophic results. Soft-real time systems allow for some deadlines to be occasionally missed with only a degradation in performance but not a complete failure.

2.2 LOCAL AREA NETWORKS FOR REAL-TIME APPLICATIONS

Real-time distributed systems are a relatively new area of research and development, and have yet to find sound theoretical foundations. The real-time concept introduces several issues not previously of concern in the design of distributed systems. It reveals the need for a basic understanding of the general properties of these systems and a major rethinking of their design mechanism. A multitude of models, such as communicating sequential processes [Hoar78], distributed processes [Hans78], the tasking model of Ada [Ich79] and graph based model [Moka83] have been proposed to analyze real-time systems. Some of the above models aimed at creating a high degree of parallelism by defining efficient interprocess communication primitives. Others attempted to cater to the timing requirements of the real-time environment. The proposed research assumes that the mechanisms to explicitly specify or enforce critical timing constraints are provided. The study focuses on the efficiency of the communication network in enforcing the timing constraints imposed by the underlying applications.

The remainder of this chapter focuses mainly on the issues related to the design of multi-access protocols for local area networks. First, we discuss the impact of the real-time concept on the design of local area networks for real-time applications. The result is a subset of design requirements that must be satisfied by any communication subsystem intended for real-time applications. Second, we study the suitability of the standard and proposed local area network protocols to handle traffic in a real-time distributed system. The adherence of these protocols to the defined subset of requirements is discussed.

2.2.1 Real-time Traffic Specification

At the highest level of abstraction, a distributed real-time system can be viewed as consisting of a set of *objects* and a set of *processes* [Insu85]. Each of these objects can be thought of as an individual component of the system. A device object, for example, represents the abstraction of attached special purpose hardware and consists of interrupt and control routines. A set of invariant properties characterizes the behavior of the system objects. An object can only change state, behave, be manipulated, or stand in relation to other objects in ways that preserve the invariant properties dictated by the system.

The states of system objects are altered by a set of *real-time processes*. A real-time process is a process that has timing constraints and whose correctness depends on the adherence to these constraints.

The set of real-time processes can be divided into two classes: *periodic* processes and *sporadic* processes. Periodic processes are ready at regular interval of times. They are characterized by a stringent deadline. Sporadic processes can be requested at any time. They come in response to external events or interrupts from the controlled systems. They are characterized by larger inter-occurrence times and less stringent deadlines than periodic processes.

Communication and synchronization of these processes is achieved through the exchange of messages. These messages may be either generated by interrupt routines which are invoked by external processes or used for communication and synchronization purposes. In order to meet the timing requirements imposed on different processes, deadlines must be assigned to different messages. Every message which exceeds its deadline is considered lost. Processes may be affected by the loss of the messages they generate.

A serious difficulty in meeting the requirements of distributed real-time applications is presented not only by the processor speed, but also by the sharing policy of the communication network which interconnects these processors. In fact, the issue of real-time control requires the establishment of new principles to aid the support of real-time performance. The concept of *response time*, defined as the time it takes a system to react to a given input, is fundamental to the design of real-time systems. However, the distribution of real-time applications to multiple concurrently executable computing resources introduces a new concept in the performance profile of the system. No longer is response time the only significant factor of the system performance. In addition, the number of lost messages impacts heavily the overall performance profile of the system. A major goal in the design of a communication subsystem in a real-time environment is to minimize the inflicted penalty by reducing he number of lost messages. The above goal is further complicated by the types of messages generated by different real-time processes. On one hand, the interaction between control computer software and the technical process requires that *periodic* and *control* messages be transmitted over the communication network. On the other hand, the need to perform control actions on stochastically occurring events in the system will require communications to support *aperiodic* messages. The characteristics of these messages vary dramatically, and are anticipated to stress correspondingly different communication architecture designs and protocols. Consequently, the communication network must be efficient to help reduce the substantial amount of overhead incurred by the synchronization of different classes of processes running on different processors. The communication subsystem must guarantee adequate performance for various timing constraints. In addition, the network must be capable of handling various traffic generated by the different sources in such a way that the specific timing constraints are satisfied.

2.2.2 Communication Requirements

Real-time applications encompass very different types of activities. In trying to determine the design requirements of real-time local area networks, the scope of research has been very broad. However, issues such as ***robustness, timing requirement, flexibility*** and ***priority mechanism*** [Lela83, RoTo81]. have dominated the design decisions of local area networks for distributed real-time systems. Following is an overview of these issues and a brief description of their relative importance in the design process.

2.2.2.1 Robustness

Robustness, defined as a combination of reliability, availability and dependability requirements, reflects the degree of the system insensitivity to errors and misinformations. Robustness is a multi-dimensional activity which must simultaneously address issues such as error confinement, error detection and masking, reconfiguration and restart. It is widely admitted that failure tolerance in real-time systems will limit the number of stations in a realistic network implementation to fewer than fifty [RaLC78]. The later restriction is reasonable for real-time systems which usually operate in a hostile environment.

2.2.2.2 Flexibility

Flexibility relates to the ease of designing and structuring a real-time network. It indicates the degree of network adaptability to a changing environment without significant disruption of the network functions. Environmental changes may be required by the need to enhance the performance of the network or the necessity to expand the basic functional services the network provides. Three types of flexibility can be achieved. ***Functional flexibility*** allows the evolution of the services required. ***Implementational flexibility*** allows an easy integration of new stations into the network. Finally, ***topological flexibility*** allows the evolution of the physical dimensions of the network by extending the current communication medium. The objective of extensibility requires a modular architecture of the system. Modularity simplifies the overall design, allows an incremental installation of the network, and reduces the complexity of its maintenance.

2.2.2.3 Timing requirements

Timing requirements refer to the type of timing guarantee provided by the network to any given station attempting to access the channel. Two types of timing guarantees can be identified: ***probabilistic guarantees*** and ***deterministic guarantees***. Access delays in the probabilistic timing are characterized by an expected value, a variance and a confidence interval. Deterministic timing ensures a predictable number of state transitions between message arrival and message departure. Therefore, deterministic access schemes guarantee an upper bound for the access time. Determinism is a crucial requirement in a real-time environment. The guarantee of an upper bound on the access time is necessary in order to meet the timing constraints of various messages. Determinism involves several parameters that need to be taken into consideration.

2.2.2.4 Priority requirements

Designing a robust and reliable network, which guarantees a finite upper bound for the channel access delays, solves only one part of the problem of how to meet timing constraints. In addition, the network must be capable of handling various traffic generated by the different sources in such a way that the specific timing constraints are satisfied. To achieve the latter goal, the network must be provided with a priority structure among the message classes, since the different classes usually have different delay requirements. Based on the priority structure, an efficient scheduling of messages can be achieved. The priority will be used to associate costs to messages, so that the network message server can determine in which order to serve messages present in a waiting queue in an *optimal* fashion. We consider that an optimal scheduling is achieved when the following conditions are satisfied [Lela85]:

(1) Finite upper bounds are guaranteed for message service times,

(2) The cost function is minimized when the timing constraints are exceeded.

In discussing implementation of prioritized schemes Rom and Tobagi [RoTo81] indicated four basic requirements for acceptability of a given priority scheme:

- *Hierarchical independence of performance:*

  The performance of a given priority class should be independent of the load of lower priority classes. This requirement ensures that increasing load of the lower priority classes of message does not degrade the performance of higher priority classes.

- *Fairness within each priority class:*

  The performance of the scheme as seen by a message of a given priority class should be fair, in terms of channel access time, among contending messages of the same priority class.

- *Robustness:*

The proper operation and performance of the priority scheme should be insensitive to errors in station information.

- *Low overhead:*

  The overhead required to implement the priority scheme and the volume of control information to be exchanged among those stations must be kept minimal.

These four requirements will be used to evaluate various priority schemes. It is clear that trying to simultaneously achieve optimal performance with respect to some of these criteria results in suboptimal performance with respect to others. Thus, a compromise must be worked out between them. Also one can notice that fairness defined for messages within the same priority class, based on a global first-come-first-served discipline, cannot be achieved efficiently in a distributed system. The overhead incurred makes such an approach prohibitive.

The need to design an efficient communication system for real-time applications motivated several research projects. The following section is dedicated to identify the issues related to the design of an arbitration scheme for local area networks. An overview of the proposed schemes is provided and their suitability for real-time traffic is discussed.

2.3 BACKGROUND

In many local area networking environments, communication among stations is usually provided by means of a unique channel. It is characteristic of this channel that only a single station can transmit a message at any given time. Therefore, shared access of the channel requires the establishment of a protocol among the network stations. A protocol is the procedures and conventions used to regiment the progression of events required for orderly, mutually understood interaction between processes [Stac80]. The difficulty in designing an effective multiaccess protocol arises from the spatial distribution of the stations. To reach a common agreement, the stations must exchange some amount of explicit or implicit information. However, the exchange of coordinating information requires the use of the channel itself. This recursive aspect of the multiaccess problem increases the complexity of the protocol and the overhead of the channel. This issue is further complicated by the absence of an instantaneous state of the system. The spatial distribution of the system does not allow any station on the network to know the instantaneous status of other stations on the network. Any information explicitly or implicitly gathered by any station, is at least as old as the time required for its propagation through the channel.

The factors which influence the aggregate behavior of a distributed multiple access protocol are the intelligence of the decision made by the protocol and the overhead involved. These two factors are unavoidably intertwined. An attempt to improve the quality of decisions does not necessarily reduce the overhead incurred. On the other hand, reducing the overhead may result in lowering the quality of the decision. Thus, a tradeoff between these two factors has to be made. A distributed multiple access protocol can potentially benefit from the globally available knowledge about the status of the critical resource. An update of the status of the transmitting station may be included in the transmitted information. The new status then becomes globally known to all to other stations in the network.

Determining the nature and extent of information used by a distributed multiple access protocol is a difficult task, but potentially a valuable one. An understanding of exactly what information is needed could lead to an understanding of its value. Most of the proposed distributed multiple access protocols operate somewhere along a spectrum of information ranging from no information to perfect information. Three types of information, *predetermined*, *dynamic global* and *local*, can be readily identified. *Predetermined* information is known to all stations. *dynamic global* information is acquired by different stations during the evolution of the protocol. *local* information is known to the individual station. The use of local information may result in a lack of coordination among stations and eventually fail to achieve the objective of the protocol. Predetermined and dynamic global information may result in perfect coordination among the stations, but usually induces a price to pay in terms of wasted resource capacity.

Over the last decade, several approaches to solve the multiple access problem were proposed. These solutions attempt, by various mechanisms, to strike a balance between the amount of information provided to the stations and the overhead in providing it.

Multiple access protocols divide broadly into two classes: *contention based* protocols and *control based* protocols.

Contention based protocols can be characterized as a partitioning process in which the set of contending stations is gradually reduced, according to some predetermined and globally known rules, until the set contains exactly one station. The remaining station successfully transmits its message. At the end the of transmission, a new partition is found. Several mechanisms have been proposed to determine the partitioning process. These mechanisms can be broadly divided into three groups [Kuro84]. The first group is based on a *probabilistic partitioning*, in which stations in the contending set randomly reschedule their transmission. The amount of coordination among different stations varies from one scheme to an other [Abra70, Abra73, KlTo75, KlSc80, KlYe78]. The second group uses an *address based* mechanism to achieve partitioning of the set of the contending stations [Cape79, Haye78, GrSH80]. At the occurrence of a collision the set of contending stations is reduced to those stations whose addresses fall within a specific range. Further collisions will reduce the range of addresses, until a single station is uniquely determined. The third group of contention based protocols use the *generation times* of messages to partition the contending set of stations. A time window is determined and only stations holding messages generated during the time window have the right to transmit. The window is recursively narrowed until it contains only one station [MeBo76].

Control based protocols are characterized by collision free access to the channel [Schw77]. Collisions are prevented by imposing an explicit or implicit ordering of the stations in the network. Explicit ordering requires a predetermined channel allocation; the channel is statically allocated to different stations on the network. Demand adaptive protocols dynamically adjust in attempt to consistently satisfy the immediate requirements of the stations. Demand adaptive protocols can further be divided into two classes: the *reservation class* and the *token passing class*. In the reservation scheme [KlSc80], stations reach a consensus about which station is next to transmit using a reservation procedure that typically precedes the successful transmission of a message. The token bus access scheme relies either on an explicit [FaNe69, Farb73, Clar78, Bux81] or implicit [Chla79] token which is passed among active stations. The active stations form a logical ring on which the token circulates. The token gives its holder the right to transmit.

Contention based schemes are simple to implement. They do not require active channel interfaces and do not rely on global information circulating on the bus. Consequently, they are robust and reliable. Control based access mechanisms reduce the degree of concurrency but guarantee a limit on the maximum time any given station must wait in order to access the channel.

Local area networks are steadily increasing their range of applications. However it is their envisioned role in supporting real-time applications which is continuously receiving increased attention. Therefore the need for priority functions becomes a crucial matter for the scheme to be responsive to real-time requirements. In addition, the protocol must guarantee adequate performance, even when the network is heavily loaded. In the next section, we briefly describe the three major standard protocols proposed to handle local area networks traffic, namely Carrier Sense Multiple Access with Collision Detection (CSMA/CD), token bus and the token ring. Note that these three standards are all defined in the data link layer, or more precisely, in the media access control (MAC) sublayer [Myer82].

2.3.1 CSMA/CD Bus Network

CSMA/CD is a random accessing scheme in which a network station transmits only after sensing an idle channel. Should the channel be busy, the station must wait until it is clear, before it can attempt transmitting again. Once it transmits its message, the station continues to listen to the channel to detect any collision with messages being sent simultaneously by other stations on the network. CSMA/CD methods ensure that collision can be detected by requiring message length to be greater than a predetermined minimum. In the event of a detected collision, all transmitting stations abort their operations and back-off for a random amount of time before attempting to retransmit. This random back-off time ensures that one station will subsequently gain access to the channel. However, the total number of times a station must try before successfully accessing the channel, is strongly influenced by such factors as the network traffic volume, the message length, and the network physical length. Consequently, the CSMA/CD based protocols perform optimally under light conditions, but the performance of such schemes is very sensitive to the traffic load. As the load increases, the scheme performance degrades significantly. The conflict that occurs due to random access can cause waste of bandwidth and most of all may result in an unbounded access delay. The last shortcoming makes the scheme unsuitable for real-time applications.

In addition to its lack of determinism, the standard CSMA/CD does not support a priority function necessary to handle different classes of traffic characterized by different timing constraints. In order to overcome most of the shortcomings of the CSMA/CD scheme, several approaches have been proposed in the literature. Some of the proposed schemes have aimed at compensating for the lack of determinism of the CSMA/CD protocol and improving its delay characteristics. A unified window access time is proposed by Juang and Wah [JuWa83, WaJu83]. The proposed protocol randomly assigns numbers to stations and resolves the collision by selecting a unique station among the contending ones in a deterministic fashion. The search procedure, based on the dynamic programming approach, is shown to be optimal. However, the procedure requires either a prohibitive amount of computation or a large amount of storage. Eswaran, Hamasher and Shedler [EsHaSh81] proposed a collision-free access protocol. The protocol uses an independent control wire to reserve the data channel. The protocol assumes that each station knows the sum of station-to-end and end-to-end propagation delays. This value is used in the reservation procedure. Although it improves the delay characteristics of the CSMA/CD scheme, the proposed protocol suffers a major drawback which involves the difficulty of adding and relocating stations. Stations addition and relocation require adjustments of the value mentioned above. Spaniol [Span79] introduced a slotted protocol, basically a TDMA, which assumes that end-to-end propagation delay is negligible. The performance of the protocol degrades considerably when the bit rate of the network is high or when the time slots are short. Tokoro and Tamara attempted to decrease the rate of collision occurrence. They introduced an acknowledgment based mechanism that enables a station to transmit an acknowledgment without collision. Messages other than acknowledgments may collide. Furthermore, the collision among messages of the same class can not be prevented completely. Therefore, this mechanism can only improve the delay characteristics to a certain extent.

Others schemes attempt to improve the characteristics of the CSMA/CD by incorporating priority functions into the protocol. Franta and Bilodeau suggested a station-based priority scheme in which different stations have different rescheduling delays [FrBi80]. Iida et al. proposed message-based priority functions via the use of different lengths of preambles to determine the class of messages than can compete for the channel [IIYO80]. Tobagi provided a linear resolution scheme to determine the current highest priority class of messages [Toba82]. A priority code resolution scheme is proposed by Ni and Li to reduce the time required in determining the current highest priority class of messages [NiLi83]. Graham [Gram81] proposed a collision resolution mechanism that uses a directional receiver to detect the direction of signals in the cable, which ensures that high-priority stations can use the bus by successively aborting other lower priority transmissions. However, time loss is incurred as the resolution of each collision requires collision window time for each retransmission. In addition it is inequitable in that the highest priority station always preempts messages transmitted by lesser stations. Zhao et al. [ZhSR87] suggest a window protocol for real-time communication which implements the "minimum-laxity-first" policy. The protocol accurately achieves its purpose so long as ties among message laxities do not occur. If two messages have the same laxity, however, the protocol uses a probabilistic approach to resolve the tie. In addition, the implementation of the protocol requires perfect synchronization of the clocks at all nodes. It is well known, however, that clock synchronization in a distributed system is a challenging issue, if not impossible to achieve [Lela83, Lamp85]. The dependence on clock synchronization can at least cause the performance of the protocol to greatly degrade.

2.3.2 Token Ring Network

In a ring topology network, the token is passed along from one station to another in one direction. Data is sequentially transferred in a bit by bit fashion from station to station on the ring. Each station on the ring introduces at least one bit delay during which it regenerates and repeats a received bit. The token could be either a free token or a busy token. A station can transmit a packet when it gets a free token and it will convert that free token to a busy token. The packet will traverse along the ring and will be removed by the sending station. At the end of a packet transmission, the station will convert the busy token back to the free token and pass the free token to the next station on the ring [DiSM83]. Ring networks provide an alternative whose throughput does not degrade significantly. The delay performance of the scheme is not sensitive to the number of stations in the ring. Further, since the signal is regenerated at each node, greater distance can be covered.

The token ring network provides a priority handling mechanism [Stro83]. In this scheme, the start-delimiter (SD) field (1-byte long) of the packet format assigns three priority mode (PM) bits to designate up to 8 different priority classes for the packet. When a station receives a token packet, it compares the priority bits of the SD field with the priority bits of the packet to be transmitted. If the comparison is favorable, the station transmits the packet; otherwise, the packet is held and the station immediately forwards the token to the next station. The SD field also has three priority reservation (PR) bits. A station can reserve its priority request in the PR field of a packet if the station's priority is higher than any current reservation request. The current transmitting station examines the PR field and releases the next free token with the new priority mode indication, but retains the interrupted priority class for later release. A requesting station uses the priority token and releases a new token at the same priority so that other stations assigned that priority can also have an opportunity to transmit. When the station that originally released the free token recognizes a free token at that priority, it then releases a new free token at the level that was interrupted by the original requests. Thus, the lower priority token resumes circulation at the point of interruption.

In general, the priority scheme adopted in the token ring network performs well. However, in some situations the requirement of hierarchical independence may not be satisfied. For example, the priority reservation information may be outdated if the new high priority packets arrive before a free token is released and after the reservation is made. In this case, the free token may be occupied by a lower priority packet. Furthermore, due to the nonpreemptive transmission, a higher priority station can do nothing but make a reservation and wait during the circulation of a busy token. The priority scheme also is not fair and favors the low priority stations located downstream and closer to the high priority stations which are mostly active. As a result, some stations might be cut off from the ring if they are located farther away from some high priority class stations having heavy traffic. The requirement of robustness may also not be satisfied. If a station which did change the token priority fails to function later, the token priority could never be brought down and this will cause starvation among low priority stations. In addition, the complexity of the token ring and the necessity to maintain an active channel interface is unsuitable for real-time applications.

2.3.3 Token Passing Bus Network

Token passing involves circulating a unique bit sequence (the token) among the network stations in a specific order. Only the station currently holding the token has access to the channel. Token bus incorporates the advantages of the token ring. Stations are connected to the bus via taps. Since the taps are passive elements, as opposed to the active interfaces of a ring, bus networks are less susceptible to failure.

Token passing guarantees access to each network station within a prescribed period of time. The highly deterministic nature of this scheme is a critical quality for distributed real-time applications. In a standard token passing protocol [IEEE83], three bits in the packet are used to represent up to 8 different priority classes (PC). However, only four priority classes are used in the current proposal. These four classes of services are synchronous (PC=6), asynchronous urgent (PC=4), asynchronous normal (PC=2), and asynchronous time-available (PC=0) [Stal84]. In this priority scheme, the lower priority packets are deferred when the network is heavily loaded. Network loading is computed at each station by measuring the time elapsed between two consecutive tokens. When the network is lightly loaded, a station passes the token and receives it again in a short period of time. As loading increases, the time for the token to return to the station increases. If the time exceeds a predetermined threshold value, low priority traffic is deferred until the network load decreases. A separate threshold value exists for each of the three lower priority classes. The highest priority class of packets can always be transmitted when the station receives the token and the token-holding timer is not expired.

The major drawback of the token based networks is the overhead involved in passing the token. This overhead has a significant impact on the performance of the network, especially in a light load situation or when the traffic distribution among stations is asymmetric. Since the token circulation path is static, the token may visit idle stations, resulting in a waste of bandwidth and an increase in the overall delay of the network. This effect becomes drastic when priority schemes are implemented. As described above, the object of the priority is to allocate network bandwidth to the higher priority class messages and to send lower priority messages when there is enough bandwidth. Consequently, a station receiving the token may not be able to transmit any messages due to the constraints imposed by the priority scheme. The token must be passed to the next station, resulting in a waste of bandwidth and a degradation of the overall performance. A closer look at the token bus priority scheme reveals also that the priority scheme lacks fairness and fails to meet hierarchical independence requirements. Since the token passing sequence is not necessarily transferred in the order of the addresses for any priority class except the one with the highest priority, the order of channel access is dependent on the traffic in other classes and different values of thresholds. In the worst case, it is possible that the packets of a lower priority class at a station get blocked, while the packets with the same priority at another station get transmitted.

We notice finally that the reliable operation of token based networks relies on the integrity of explicit information such as a unique token or, on the integrity of the active stations. In addition to degrading the overall performance of the network, improper behavior of the stations destroys the deterministic properties of the network. The resolution of exceptional events, such as the addition of a new station to the logical ring, may not be collision free, violating thereby the main objective of the protocol, collision elimination. Similarly, it is difficult to predict what impact exceptional events, such as the election of a new control unit or the recovery from a token loss, might have on access delays.

In order to overcome most of the short-comings of the previous schemes, several approaches have been proposed in the literature. Some of these approaches have aimed at eliminating the physical circulation of the token on the network [LiFl82, UlWA81, ToBF82, FiTo82]. Instead, an implicit token was used to determine the next station allowed to transmit. Implicit passing of the token considerably improves the overall robustness of the network. However, under light loads, the performance of these schemes remains poor. In addition, little effort has been made in attempting to improve the characteristics of the priority scheme.

The objective of this work is to investigate efficient and effective strategies to handle messages communication in real-time environment. The need for an efficient priority scheme at the data link layer is crucial to the implementation of an efficient scheduling policy at the inter-process communication layer. A novel access arbitration scheme is proposed in the next chapter. The scheme attempts to preserve the main advantages of the CSMA/CD and token bus schemes. Above all, the scheme guarantees a more efficient way of handling different priority classes by implementing a priority mechanism that satisfies closely the basic requirements of a general priority scheme.

2.4 CONCLUSION

In this chapter we identified the basic design requirements of distributed real-time systems. The fundamental problems introduced by these systems and the solutions they require have been discussed. The main focus of the chapter, however, was dedicated to discussing issues related to the design of multi-access protocols for local area networks intended to handle distributed real-time systems. The basic requirements that must be satisfied by these communication protocols were provided and the adherence of the standard protocols to these requirements was discussed.

In our discussion of the CSMA/CD based schemes, we have observed that the probability of collision increases as the load on the channel increases, leading to poor channel throughput and worst of all to an unbounded access delay. In addition, we noticed that the protocol does not handle different message priority classes. The absence of a guaranteed upper bound on the access time and the nonexistence of a priority mechanism to handle different classes of messages are undesirable features of any communication protocol intended for real-time application.

On the other hand, we have argued that the performance of a token-based access mechanism decreases in a light load situation, due to the overhead generated by the token traveling among idle stations. In addition, the priority scheme provided by these protocols, rather a bandwidth allocation, violates the hierarchical independence requirement of a priority scheme.

We conclude our discussion by observing that from a performance point of view, it is desirable to have a protocol which achieves good performance over the entire range of channel utilization. Moreover, in order to provide acceptable performance in real-time environments, an efficient priority scheme is required. Therefore, a more efficient protocol for handling messages with different priority classes would combine the advantages of both CSMA/CD and token-passing bus protocols, but avoid their shortcomings. The above observation lead to the proposed protocol described in the subsequent chapters.

Chapter 3

A Priority Based Resolution Multiaccess Protocol For Real-Time Systems

In the previous chapter, we set forth the basic requirements of real-time applications, and described the basic functionality and architecture of multiaccess protocols for distributed real-time systems. In our review of the proposed schemes to handle distributed real-time applications, two medium access schemes, CSMA/CD and token-passing based protocol, received much attention. Our interest in these schemes stems from their relatively simple topology. In addition, as stated in the previous chapter, CSMA/CD based schemes provide a good throughput-delay performance at low channel utilization even for a very a large number of stations. Token-passing based protocols guarantee an upper bound on the access delay and can achieve high channel throughput even when the channel untilization is high. However, neither CSMA/CD nor token-passing based protocols fully satisfy the requirements of an access mechanism designed to handle real-time traffic.

In this chapter, we describe a channel access scheme which attempts to preserve the main advantages of both CSMA/CD and token passing bus schemes and remedy their aforementioned shortcomings.

The first part of this chapter is devoted to describing the design requirements of the proposed protocol. The framework of station communication is then introduced and the basic data structure required for the proper functioning of the protocol is described. In the second part of the chapter we introduce the concept of an ideal protocol as a pictorial aid for the design of multiaccess protocols. We then provide a description of the protocol by tracing its basic operations. Next, we formally describe the behavior of the protocol using a state tabulation method. The last part of this chapter is dedicated to discussing issues related to the implementation of the network priority list, a key data structure for the proper functioning of the protocol.

3.1 DESIGN REQUIREMENTS

Prior to detailing the proposed protocol, we describe the network structure and the design requirements for the protocol implementation.

3.1.1 Network Architecture

It is assumed that the network has at most N processors, referred to as stations. In a real-time environment, the need to reduce failure rate and achieve acceptable protocol performance drive realistic network implementors to limit the stations connected to the network to a relatively small number. The set of connected stations uses a single channel for communications. At a given time only one message can be successfully transmitted over the channel. When more than one station attempts to transmit simultaneously, a collision occurs and the information is lost. Note that the amount of time during which a collision may occur is twice the maximum round trip end-to-end propagation delay. The broadcast nature of transmission allows stations to monitor the activity of the channel and either recognize a successful transmission, detect a collision or sense the channel idle. The ability of stations to detect message collisions and subsequently abort transmission of their messages ensures that the channel is not wasted by the transmission of a colliding message.

The network stations are ordered so as to form a logical ring. Every station in the logical ring is assigned a unique *logical identity* (LID), s, where $0 \le s \le N-1$. We assume these identities are assigned when the stations join the network for the first time. The assignment of these identities can be achieved by some control or management procedures. The scheme makes no assumptions regarding the mapping between a station's LID and its physical address, or the policy and the order according to which an identity is assigned. Stations are not required to memorize the logical identity of their successors or predecessors in the logical ring. Furthermore, stations are not assumed to have any knowledge of the positions of other stations in the logical ring or the distances that separate them.

The logical ring can either be *static* or *dynamic*. In a static configuration, the logical identity infers a predetermined sequential ordering of the stations, so that the position of any given station in the ring is fixed.

In a dynamic ring configuration, stations can voluntarily drop out of the ring, thereby relinquishing their current position in the logical ring and reducing by one the size of the ring. The size of the logical ring may also be reduced whenever the failure of a station previously part of the ring is detected by other stations in the logical ring. A reconfiguration of the ring is then required and the failing station is temporarily removed from the ring. Similarly, a ring reconfiguration is necessary whenever a station not currently part of the logical ring wishes to join the ring. At the completion of the reconfiguration process, the station is assigned a position signifying its reintegration into the logical ring. In a dynamic configuration, the current position of a station in the logical ring will be referred to as its *station index*. Note that in a dynamic ring configuration, the station logical ID and the station index are not necessarily the same.

3.1.2 Performance Issues

Protocols require the use of channel for explicit exchange of control information in order to achieve some form of coordination. The fraction of the channel used for successful message transmission, as opposed to control information and message collisions, is known as the *effective utilization* of the channel [Lam75]. The maximum value of the effective utilization over all possible traffic loads is known as the *capacity* of the protocol. Several aspects of the multiple-access environment and the behavior of the protocol itself influence its capacity [Yemi79]. This capacity characterizes the performance behavior of the network. Another measurement to be considered in the performance evaluation of a network is the *message delay*, defined as the time interval between the instant at which a message is delivered to the source station and the instant at which that message is successfully received by the destination station. The message delay consists of an *access delay* and a *communication delay*. The access delay of a message is the time required for the ready message to successfully gain entry to the communication channel. The communication delay of a message is the time spent on the channel and is determined by the channel transmission rate and the message length. Throughput and delay characteristics have been the most frequently cited performance measures in the literature. The trade-off between the access delay and the throughput reflects the effect of an increasing message generation rate on the average message delay.

In a real-time environment, however, an important performance consideration is the percentage of messages that are received at the destination station before the expiration of their deadlines. If the delay of the message is exceeded, the message is considered lost regardless of whether it is ever received by the destination station.

3.1.3 Station Design Issues

The purpose of the following section is to briefly introduce the framework of station communication in a local area network. A description of the basic data structure required in the proposed the protocol is also provided.

3.1.3.1 Inter-layer Interfaces

The framework for communication consists of many components with complex interaction between them. This characteristic makes the task of communication in a truly cooperative way between applications on different processors too complex to be handled as a single unit. The problem must be decomposed into manageable parts. Hence, a structure or architecture that defines the communication tasks is required. This line of reasoning led to the development of different models and architectures to define the communication system. A widely accepted structuring technique is *layering*. The purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how offered services are actually implemented [Zimm80].

In the proposed protocol, we assume that higher levels of the communication system establish an ordering on the set of messages generated by the network stations. The scheduling strategy used by higher layers may be based on a *dynamic* priority scheme. In this scheme, messages may be ordered increasingly according to the value of their deadlines. Based on the timing constraints, higher layers of the communication protocol assign an integer to the out-going message. This integer defines the *priority* of the message. A message will be assigned a high priority if its deadline is near and will be assigned a low priority if its deadline is far in time. We assume that the priority carried by the messages ranges between $[1,m-1]$, where 1 is the lowest priority and $m-1$ is the highest priority.

3.1.3.2 Basic data structure

The primary services provided by the low level layer to to the higher layers are transmission and reception of messages according to their priorities. In order to achieve this purpose, each station in the network maintains a *local transmission queue* (*TQ*) which holds outgoing messages ordered according to their priority classes. Messages of the same class are ordered according to their arrival times.

At any instant, the *priority* of a station is defined as the current highest message priority in its *TQ*. We will use $p(s)$ to denote the priority of station s, $0 \leq s \leq N-1$. Note that the value of $p(s)$ is subject to change due to the arrival of newly generated messages or the departure of old ones. When the *TQ* of station s is empty, its priority is set to 0; this is is referred to as the *base priority*. Therefore, $p(s)$ will always range between 0 and $m-1$. Finally, we define the *network priority*, p, to be the current highest priority among all stations. Thus, at any instant $p = max\{p(s) \mid 0 \leq s \leq N-1\}$.

In addition to its transmission queue, each station maintains a data structure called the *Network Priority List* (NPL). Each NPL has N entries, one for each station. Each entry has two fields, *NP* and *LINK*. The field $NP_i(s)$ denotes the most recently gathered information about the priority of station s ($0 \leq s \leq N-1$) by station i ($0 \leq i \leq N-1$). Consequently, $NP_i(s)$ will be updated after each successful transmission of a frame by station s. The field $LINK_i(s)$ is used to maintain the entries in sorted order according to the value of $NP_i(s)$. Due to the broadcast nature of the shared bus, all active stations are able to receive any information successfully placed on the channel. In general, the content of local NPL's will be consistent over all stations. Thus, $NP_i(s) = NP_j(s)$ and $LINK_i(s) = LINK_j(s)$ for all $i \neq j$ and will be denoted as $NP(s)$ and $LINK(s)$, respectively.

In order to implement a priority mechanism in such distributed systems, we need to address three basic issues, namely to :

- identify the instants at which the network priority , *p*, is assessed,
- design a mechanism by which the value of p is identified,
- design a mechanism which guarantees the right of channel access to a unique station among all contending ones.

The network priority list will prove to be useful in achieving the above task. However, since the NPL is accessed frequently, an efficient mechanism for its implementation and updating is required. Issues related to the implementation of the network priority list will be considered later in this chapter.

3.2 PRIORITIZED HYBRID VIRTUAL TOKEN-PASSING PROTOCOL

As stated previously, the objective of any access mechanism supporting priority functions is to identify the network priority. The efficiency of the scheme depends strongly on the overhead incurred in the attempt to assess the network priority and resolve conflicts which arise when more than one station holds that priority. Achieving this aim has varied from one scheme to another. However, there is a great deal of conceptual overlap between these schemes. In this section, before describing the proposed protocol, we attempt to define an ideal scheme which supports priority functions. The understanding of the priority ring concept this scheme introduces will make the design, implementation and comparison of priority based multiple access mechanism more tractable.

3.2.1 Ideal Priority Scheme

Conceptually, the priority classes defined by the access mechanism can be represented as a set of logical sub-rings, $\{R_i, 0 \le i < m\}$, where m denotes the highest priority class. Each logical sub-ring corresponds to a distinct priority class. The stations in ring R_i, $i > 0$, are those stations who currently have a priority i message to transmit. These stations will be referred to as *ready* stations. Stations with empty *TQ*'s belong to

R_0 and will be referred to as *idle* stations. Let $r(i)$ denote the number of stations in R_i. Formally, station $s \in R_i$ if and only if $p(s)=i$. Therefore, we have $r(i)=|\{s, \textit{ such that } p(s)=i\}|$. Furthermore, R_t is the current highest priority class sub-ring if and only if $r(k)\neq 0$ and $r(i)=0$ for *all* $i > k$.

An ideal scheme which supports m different priority classes will always circulate the token, or access right, around the current highest priority class sub-ring, class-t sub-ring. The token implicitly travels from one ready station to another. The order in which the token visits these ready stations is such that the station whose message arrived first is served first, the station whose message arrived second is served second, and so on. Two circumstances can interrupt the circulation of the token in the current highest priority sub-ring. First, the sub-ring may become empty. As stations in the ring transmit their messages, they may drop from R_t to lower priority sub-rings, depending on the priority of the messages left in their *TQ*'s. If all the stations in the sub-ring are serviced and drop to lower priorities, then the token moves to the non empty sub-ring of highest priority, the new current highest priority ring. Second, if a message arrives with priority greater than t, say t', then ring $R_{t'}$ is no longer empty, and it becomes the new current highest priority sub-ring. The token interrupts servicing stations in priority class-k sub-ring and travels up to service stations in priority class-t' sub-ring. At the completion of its sojourn at priority class-t' sub-ring, the token implicitly travels down to the next highest priority sub-ring, skipping all empty priority class sub-rings.

An illustration of the ideal access mechanism is presented in Figure 3.1. We assume m different priority classes are allowed. After serving priority class-k sub-ring, Figure 3.1-(a), the token drops to class-r sub-ring, Figure 3.1-(b). However, the arrival of priority $m-1$ message forces the token to interrupt its sojourn at the class-r sub-ring and travel up to class-$m-1$ sub-ring, Figure 3.1-(c). At the completion of the service at the latter sub-ring, the token resumes service at priority class-r sub-ring, Figure 3.1-(d).

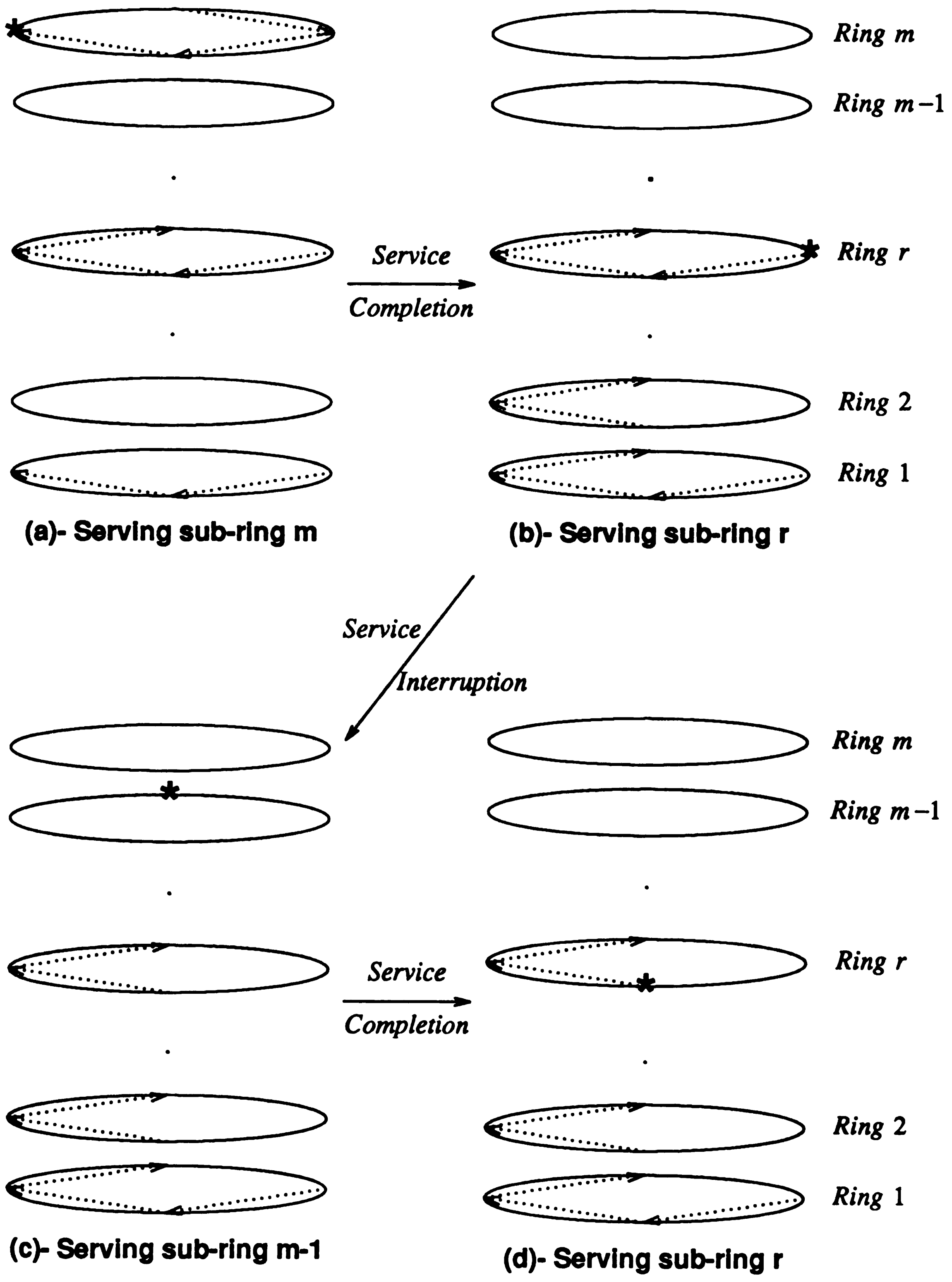

Figure 3.1 Ideal Priority Scheme

It is clear that the ideal scheme is responsive to different priority classes. The scheme ensures that the performance of higher priority class messages is insensitive to the load exercised on the channel by lower priority classes. In addition, the scheme allows different stations within the same priority class sub-ring to contend with equal right. Therefore, this scheme satisfies the criteria of hierarchical independence and fairness. However, the real implementation of such a scheme would require a certain amount of overhead. This overhead includes :

- determining the current highest priority sub-ring,
- handling the token circulation among ready stations in the current highest priority sub-ring and between different priority sub-rings.

The proposed multiaccess protocol is an attempt to approach the performance of the above idealized channel access scheme with minimal implementation overhead. Conflict free access of the channel by stations at a given priority is realized by establishing a virtual sub-ring among stations currently holding a message of that priority. Stations currently part of the logical sub-ring transmit their messages in a cyclic fashion. After serving all ready stations of the current highest priority, the token implicitly drops to the next highest priority level, bypassing all empty priority sub-rings. A contention-based approach causes the token to travel from the current priority level to a higher priority level. Contention resolution is resolved in a deterministic fashion. A description of the protocol follows.

3.2.2 Protocol Description

Stations on the network are always in one of three states *active*, *joining* or *inactive*. Active stations are those which are eligible to transmit. They may be either *idle*, if they have no messages to send (an empty *TQ*), or *ready*, if they have at least one message they wish to transmit. Depending on its priority, a ready station belongs to a sub-ring of that

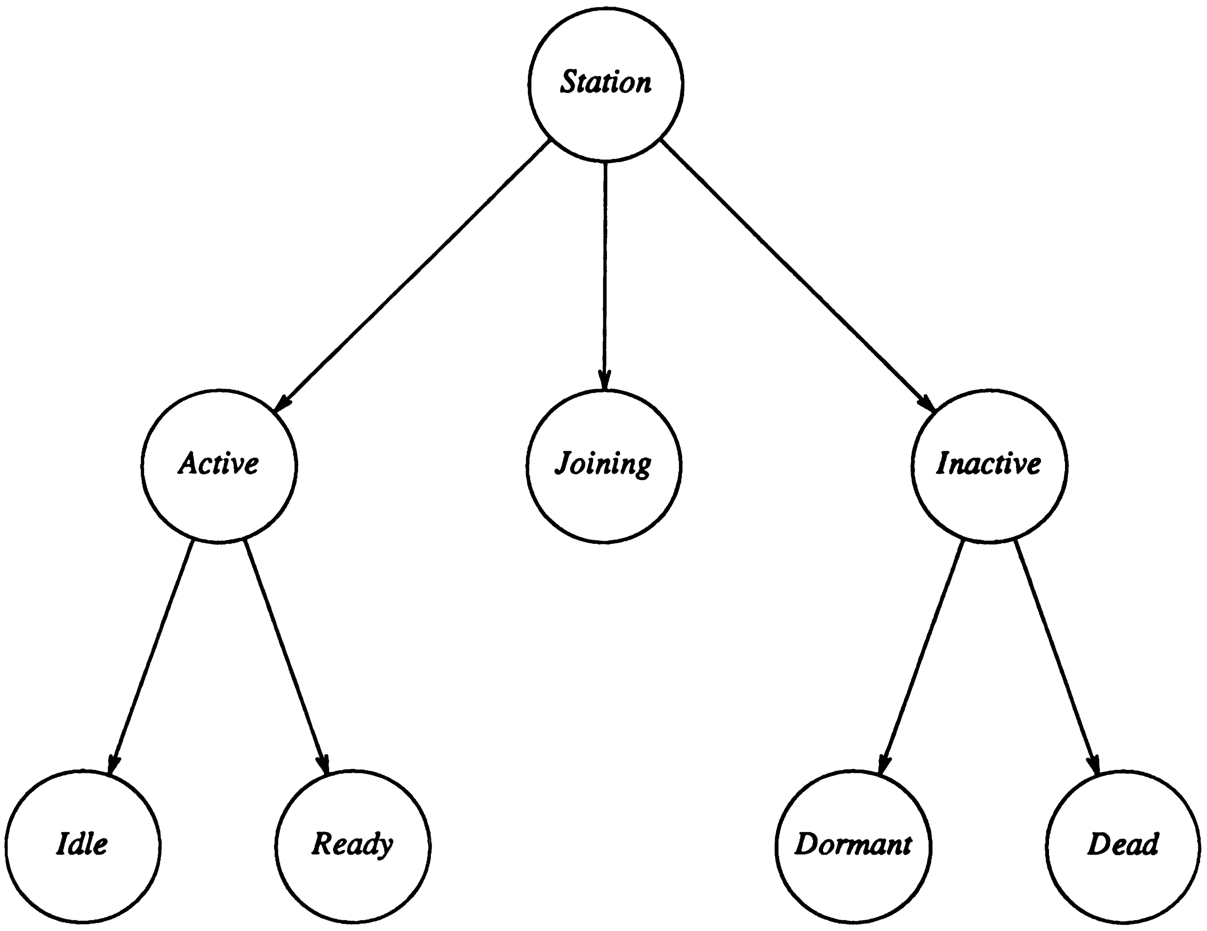

Figure 3.2 State Diagram of a Station

priority level. *Inactive* stations are those which have failed. Until its failure is generally acknowledged, the station is considered *dormant*. Once its failure is generally acknowledged, a dormant station is considered *dead*. *Joining* stations are those stations which are attempting to become active. The three station states and their further subdivisions are indicated in the stations state diagram shown in Figure 3.2.

Both active and joining stations are able to monitor the activity of the channel and detect a successful transmission, an idle channel, or the occurrence of a collision.

Associated with the protocol is an assumed structure for the information as it appears on the transmission channel. For the transmission of data packets, this entity is called a *frame* [Stal83]. However, in the following discussions, the terms frame and message will be used interchangeably. The general frame structure of the proposed protocol is depicted in Figure 3.3. The header of the frame includes fields that contain the physical address of the destination station, and the physical and logical addresses of the source station that originated the information. The next set of fields defines the control information necessary for different stations in the network in order for these stations to properly execute the protocol. Definition of these fields should become clear as we proceed with the description of the access mechanism. Note that the information carried by a successfully transmitted frame is known to all active and joining stations. The access mechanism makes extensive use of this inherent capability.

To provide for all basic operations, the protocol defines three types of frames: *Data frames*, *NPL-request frames*, and *NPL frames*. The type of the frame is identified by the value of its T field (Figure 3.3). The "data" portion of the Data frames is usually variable in length , up to a certain maximum, and contains information that the sender is transferring to the receiver. Within the data field, full transparency is provided, in the sense that any arbitrary sequence of values may appear.

NPL-request frames carry requests for the contents of the network priority list, in addition to their data unit. In order to become an active station, a station willing to join the network must obtain the content of the network priority list. The joining station manifests this desire by sending an NPL-request frame. The NPL-request frame will eventually be honored by an active station which sends an NPL frame. In addition to the station data unit, the NPL frame carries the content of the station's NPL. The additional information in the NPL frame comes in response to a previous NPL-request frame.

SD	DA	SA	S	P	R	RP	QP	NA	T	NPL	Data	FCS	ED

SD:	Start delimiter
DA:	Destination station address
SA:	Source station address
S:	Source station logical identity
P:	Priority of the message being transmitted
R:	Repeat bit
RP:	Reserved priority of the transmitting station
QP:	Current highest priority of the Network Priority List
NA:	Current number of ready stations (optional)
T:	Frame type field
	00: The transmitted frame is a DATA frame with the same priority as the reserved priority
	01: The transmitted frame is a DATA frame with a higher priority than the reserved priority
	10: The frame is an NPL-request frame
	11: The frame contains NPL information
NPL:	Content of the NPL (exists only when T=11)
Data:	Data Unit
FCS:	Frame Check sequence
ED:	End delimiter

Figure 3.3 Frame Format Description

At any instant, each station of the network belongs to a priority class sub-ring. Note that the value of $NP(s)$ indicates that station s is in the priority class-$NP(s)$ sub-ring. The ordering of station s within the priority class-NP subring is determined by the value of $LINK(s)$. Let $np = \underset{s=0,N-1}{Max} NP(s)$. Note that np represents the highest priority globally known by all stations in the network and does not necessarily represent the network priority. The station at the front of the class-np sub-ring is potentially the ***next virtual token holder***. This station will be referred to as station nv.

Before presenting a formal specification of the protocol, we first sketch it by describing its basic operations.

The type of frame currently entering the channel dictates what actions the sending station, the other active stations, and any joining stations may perform. The protocol specifies three different procedures depending on whether the outgoing frame is a data frame, an NPL-request frame, or an NPL frame.

Let *v* denote the station currently transmitting the frame and, thus, the current virtual token holder. If the outgoing frame is a data frame, station *v* fills the NP-field with the value of the current highest priority of its local transmission queue and fills the field QP with the current highest priority value of its NPL. All active stations, in the error free case under discussion, receive this message and record the value of RP as the reserved priority of station *v* in the entry of $NP(v)$ of their local NPLs. At end of the updating procedure, station *v* attempts to implicitly pass the token to *nv*, the station currently at the front of the updated NPL.

Knowing it is the next implicit virtual token holder, station *nv* attempts to transmit its highest priority message at the end of the current transmission. If the attempted transmission carries a priority higher than the known reserved priority NP(*nv*), station *nv* will signal it to other stations by setting the field T to 01. At the same time, all stations having a message with priority higher than NP(*nv*) will attempt to revoke the right to transmit from station *nv* by transmitting their own messages. Notice that the set of revoking stations may contain not only active but also joining stations. Prior to its attempt to join the ring, a joining station must listen to a full frame or sense the channel idle for a predefined time period. Similar to any active station, a joining station compares its current local priority with the network priority, *np*, carried by the transmitted frame, i.e., the maximum of the QP and the RP fields. If the current station priority is greater than *np*, the joining station has the right to revoke the channel access and will do so by attempting to send its message at the end of the current transmission.

If the set of revoking stations is empty, station *nv* will successfully transmit its message and implicitly become the new virtual token holder. However, if at least one station revokes the virtual token, transmission attempts will interfere and a collision will occur. Upon the detection of noise over the channel, the previous virtual token holder, station *v*, concludes that the attempt to pass the virtual token has failed and implicitly declares the opening of a *priority assessment phase* (PAP). The purpose of the PAP is to exclusively identify, among the revoking stations, the station currently holding the highest priority. Only stations revoking the right to transmit participate in the PAP. Given the current number of active stations, its current priority, and its position on the virtual ring, each revoking station generates a *contention parameter* and then executes the priority assessment protocol to be described later. The remaining stations, having messages with priority less than or equal to NP(*nv*), will not compete for the channel but will monitor the channel activity during the PAP in order to to update their NPLs. At the completion of the PAP, the station with the highest priority and "closest" to the last virtual token holder, the station *v*, will be selected to access the channel and become the new virtual token holder.

If a joining station is selected at the end of the PAP, it transmits its message and expresses its desire to obtain the NPL information by setting the field T to 10 in the outgoing frame, designating it as an NPL-request frame,. The field QP of the outgoing frame does not carry any meaningful information. This request will be honored by the next active station to hold the token. Note that when an NPL-request frame is detected, no station is allowed to revoke the right to transmit; the next virtual token holder will immediately honor the request. All stations willing to join the network are then able to record the required NPL information and simultaneously enter the active state. However, if an active station is selected at the end of PAP, all joining stations will reiterate their attempts to join at the end of the next successful transmission.

Finally, if the outgoing frame is an NPL frame, it can only come in answer to a previous NPL-request frame made by a joining station which has just successfully transmitted. This frame carries the content of the NPL, allowing all joining stations to simultaneously record the NPL information and update their respective local copies of the NPL. These joining stations then become active and are thereafter able to correctly execute the protocol.

During the initialization phase or when the NPL becomes empty, the network priority is reset to its base priority and all stations are in the idle state. In this state, all stations are allowed to attempt transmission immediately after the arrival of a newly generated message. If the attempt is successful, the transmitting station will become the current virtual token holder. However, if the attempt is unsuccessful due to the occurrence and detection of a collision, a PAP is declared and all competing stations move into the PAP. If a joining station detects that the channel has been idle for a predefined period of time, it assumes the network is empty and attempts its frame transmission immediately.

We have mentioned that a revoking station must have a higher priority than the implicit virtual token holder. This restriction is necessary to avoid spurious contention for the channel. However, under this scheme a situation exists where, by successively reserving the channel, the current virtual token holder may prevent all other stations with the same priority from competing for the channel. To avoid this situation, an additional restriction is required; every time the current token holder attempts to reserve the same priority as the priority of the message being sent, it sets the repeat bit R. When set, this bit signals that the station has already sent a message of the indicated priority. This will allow those stations, which are currently holding the same priority but did not have the opportunity to make a reservation, to revoke the right to access the channel.

The failure of the implicit virtual token holder introduces an exception to the previously described protocol. This situation can be identified after the channel remains idle for one slot time or when another station currently holding a message of a higher priority than the NP(*nv*) successfully transmits its message immediately following the previous successful transmission. All active stations then assume the failure of the implicit virtual token holder. Each station updates its local copy of the NPL. The station at the top of the updated NPL becomes the next virtual token holder. Notice that if more than one station attempts to revoke the right to transmit, the failure of the implicit virtual token holder remains undetected and the failing station will remain dormant.

3.3 PROTOCOL FORMAL SPECIFICATION

A State-Condition-Action tabulation method (SCA) was proposed [LGKN86] to describe distributed algorithms and access protocols. A protocol can be characterized by a number of states. The main idea behind the SCA tabulation method is to associate a table with each state of the protocol. The table consists of a *current state row*, a set of *condition rows*, an *action row*, an *action status row* and a *next state row*. The table describes the action undertaken by a station under all possible conditions. Given a current state, the satisfaction of a given set of conditions triggers the appropriate actions to be performed. The result of these actions determines the next state. In the table, the satisfaction of a set of conditions is checked by ANDing all row conditions in order to determine a unique action-column. The station then performs the set of actions described in the corresponding action-entry of the table before moving into the state described by the next state entry of the same action-column.

Depending on its current state, a station can view the protocol in one of three possible states: End Of Transmission state (EOT), Joining Attempt (JA) state, or a Priority Assessment Phase state. At the end of a successful frame transmission, all active stations move into EOT state. Joining stations, however, move into JA state. Note that the joining stations are allowed to move into the JA state since they heard a full frame over the channel. Depending on the information carried by the last successfully transmitted frame, active and joining stations determine the set of actions they need to undertake and the next state they must move into. The set of conditions and the resulting actions for the EOT state and the JA state are described in the SCA tables shown in Figure 3.4 and Figure 3.5, respectively.

To illustrate the SCA tabulation mechanism, we detail the behavior of an active station and a joining station as described in the corresponding SCA table. At the end of transmission, all active stations move into the EOT state. If the previous frame was an NPL-request frame (Fflag is set), the current token holder will honor the request in the next transmission. All remaining stations will refrain from competing for the channel. The successful transmission of the NPL-frame will lead all stations into the EOT state. If no NPL-frame was requested (Fflag is unset), the virtual token holder will attempt its transmission. All stations currently holding a message of priority higher than the current network priority list ($p(s) > \mathrm{NP}(nv)$), will revoke the right to transmit from station nv.

The same mechanism applies to describe the SCA table of the JA state. At the end of transmission, if the joining station recognizes an NPL-frame (T=11), the station updates its NPL and sets its status to active (Action). If the transmitted frame is DATA-frame (T≠1), the joining station compares its current priority to the network priority globally known by all active stations (Cond-4). Depending on the outcome of the comparison, the joining station decides either to attempt its frame transmission or postpone it for later. Finally, if the joining station detects an idle channel (Cond-1), the station transmits its frame and sets its state to active. Note that a joining station is required to monitor the channel for $\lceil \log_2(N) \rceil$ slots, during which no collision occurs before it concludes that the channel is idle. This requirement ensures that a joining station never disturbs a priority assessment phase.

Current state	EOT : End of Transmission										
Cond-1	Virtual Token Holder (s = *nv*)					Non Virtual Token Holder (s ≠ *nv*)					
Cond-2	Fflag = False				Fflag = True	Fflag = False					Fflag = True
Cond-3	p(s)>NP(*nv*)		p(s)=NP(*nv*)			p(s)>NP(*nv*) or p(s)=NP(*nv*) & R set		p(s)<NP(*nv*) or p(s)=NP(*nv*) & R reset			
Action	• Set T=01 • Transmit highest priority frame		• Set T=00 • Transmit highest priority frame		• Set T=11 • Transmit NPL Frame	• Set T=00 • Transmit highest priority frame		Monitor channel			Monitor channel
Action status	Col	Suc	Col	Suc	Suc	Col	Suc	Col	Suc	Ide	Suc
Next state	PAP	EOT	PAP	EOT	EOT	PAP	EOT	PAP	EOT	EOT	EOT

Figure 3.4 The SCA table describing the state EOT

Depending on the status of the action undertaken (Action Status), active and joining stations determine the next state. Note that in the action status row, *Suc*, *Col*, and *Idle* represent, respectively, successful transmission, collision, and idle status of the channel. Active and joining stations move into a PAP state at the occurrence of a collision. The formal description of this state is presented in Figure 3.6. In this SCA table, the parameters Low and L_i represent the lower bounds of the initial and current priority assessment intervals, respectively. A full description of the algorithm stations use to compute and update the values of these bounds is provided in the next chapter.

Current state	JA : Joining Attempt						
Cond-1	Complete Frame is heard					Idle Channel	
Cond-2	Fflag = True	Fflag = False					
Cond-3		T=11	T≠11				
Cond-4			$p(s) \geq NPL(nv)$		$p(s) < NPL(nv)$		
Action		• Update NPL • Status=Active	• Set T=10 • Transmit highest priority frame			• Transmit highest priority frame • Status=Active	
Action status			Col	Suc		Col	Suc
Next state	JA	EOT	PAP	JA	JA	PAP	EOT

Figure 3.5 The SCA table describing the state JA

Current state	PAP : Priority Assessment Phase										
Cond-1	L_i = low	L_i > low									
Cond-2		Active					Inactive				
Cond-3		cp > L_i		cp ≤ L_i			cp > L_i		cp ≤ L_i		
Action		• Set T=00 • Transmit highest priority frame		Monitor channel			• Set T=01 • Transmit highest priority frame		Monitor channel		
Action status		Suc	Col	Suc & T=01	Suc & T≠01	Col	Suc	Col	Suc	Suc	Col
Next state	EOT	EOT	PAP	EOT	PAP	PAP	EOT	PAP	EOT	PAP	PAP

Figure 3.6 The SCA table describing the state PAP

3.4 NETWORK PRIORITY LIST DESIGN ISSUES

A realistic implementation of a channel access mechanism must place heavy emphasis on such constraints as efficiency and suitability to a particular technology or architecture. In this context, several important properties of the protocol must be considered. However, since the description of the protocol provided above is only intended as the primary specification of the protocol behavior, we will only emphasize the implementation issues of the main data structure required by the protocol, namely the network priority list.

3.4.1 Implementation Considerations

The Network Priority List (NPL) is the main data structure on which relies the proper functioning of the proposed protocol. The information carried by the NPL is crucial in order for the stations in the network to correctly execute the protocol. Each station maintains an NPL which is accessed and updated frequently. Therefore, a *reliable*, *robust* and *efficient* implementation of the required data structure becomes crucial to the performance of the protocol. A data structure is said to be reliable if failures do not seriously impair its satisfactory operation [PAR75]. Therefore, reliability does not mean freedom from errors and faults, but tolerance against them. Robust data structure contains redundant information which allows erroneous changes to be detected, and possibly corrected [TaMoJ80]. Efficiency refers to the ease of accessing and updating the data structure.

The proposed approach implements the NPL as a set of doubly linked lists. Each list links all NPL entries of the same priority. Stations linked to the same list belong to the same sub-ring. Each element in the structure consists of three memory locations. The first contains the priority reserved by the corresponding station. The second and third locations contain respectively a pointer to the next station and the previous station in the priority class sub-ring. This method of storage ensures that the operation of adding and deleting stations from the sub-rings can be accomplished in a bounded amount of operations, limiting thereby the access time. However, these operations assume a prior knowledge of the location where the new item is to be inserted or an old item to be deleted. In order to achieve this goal and keep the computational complexity of handling a sequence of insert and delete operations to a reasonable amount of time, we use, for each priority class, two additional pointers. Each of the two pointers gives the location of the current front and rear of the corresponding priority class list, respectively.

The implementation of the NPL as a double linked list is illustrated in Figure 3.7. In this figure, we show the data structure required to handle N stations and four priority classes. The pointer head points to the current highest priority class globally known by the stations in the network. Top (p) and Bottom (p), $0<p<4$, give the index of the first and last station of priority class-p sub-ring, respectively, and obviate the need to search the entire list to find these elements.

The procedure shown in Figure 3.8 describes the steps required to insert a new station in a specific priority class sub-ring. The procedure requires the name of the station to be added and the priority class sub-ring.

To remove a station from a given priority class sub-ring, the procedure described in Figure 3.9 could be used.

After a successful transmission, the next virtual token holder can be identified as described in Figure 3.10.

It is clear that the translation of these operations into an executable code would result in an execution time that is independent of the size of the list. Note also that this structure allows the manipulation of the priority class sub-rings in a manner which enforces fairness among stations with the same priority. It causes stations to be inserted in a first-in-first-out basis within a priority class. The insertion order dictates the order in

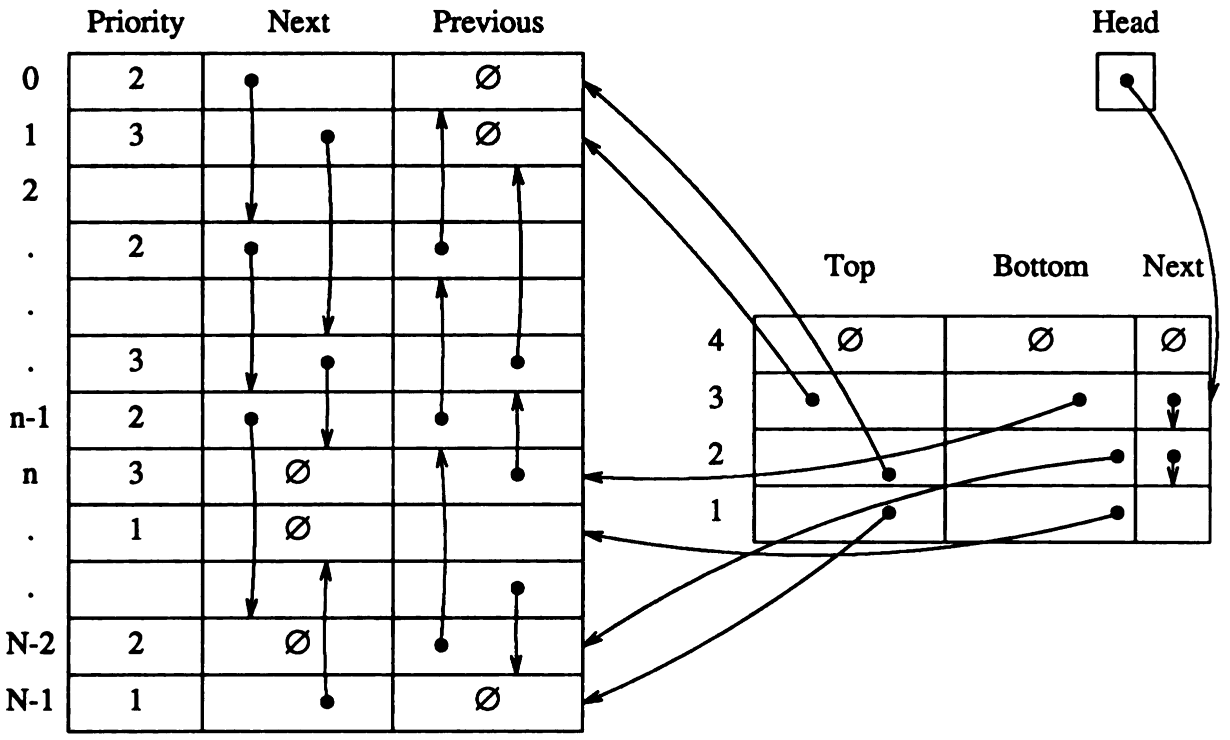

Figure 3.7. Network Priority List

which these stations are allowed to access the network.

It has been reported that the doubly linked storage structure is robust, essentially because it has two independent and disjoint sets of pointers, each of which may be used to reconstruct the entire list. However, robustness can be further improved using the ***modified double linked list*** proposed by Taylor *et al* [TaMB84,Parn75]. The modified structure is similar to the double linked list, except that the backward pointers point to the second preceding node rather than the immediately preceding one. The storage required by the modified structure is the same, but an additional step is required when inserting a new item in the list. The modified structure suggests greater detectability. However, the increased complexity of the update routines may make such structures undesirable.

```
Procedure INSERT(station, ring)
Begin
    Next[Bottom[ring]]:= station;
    Previous[station]:= Bottom[ring];
    Bottom[ring]:= station;
End
```

Figure 3.8 Procedure Insert

```
Procedure REMOVE(station, ring)
Begin
    Next[Previous[station]] := Next[station];
    Previous[Next[station]] := Previous[station];
end
```

Figure 3.9 Procedure Remove

```
Function Next_Vrtl_Tkn_Hldr : integer;
Begin
    Next_Vrtl_Tkn_Hldr:= Top[Head];
    Top[Head]:= Next[Top[Head]];
    return
End
```

Figure 3.10 Procedure Next_Vrtl_Tkn_Hldr

3.5 CONCLUSION

In this chapter we have described a protocol to handle traffic with different requirements, such as the traffic generated by distributed real-time applications. We first introduced the concept of an ideal protocol to handle different classes of priority messages. We then introduced the virtual token protocol as an attempt to approach the performance of the ideal scheme with minimum channel overhead. A description of the protocol operations by tracing its basic operations was provided and the major issues involved in the design of the protocol were discussed. A description of the required data structure to handle the basic operations of the protocol was provided and the issues related to their implementations were discussed.

The major contribution of the proposed protocol is its efficient priority handling mechanism which is not available in the CSMA/CD scheme and poorly provided in the token-passing bus. The proposed priority mechanism more closely satisfies the requirements of a general priority scheme than currently existing methods. This allows the protocol to handle real-time requirements more efficiently.

In the next chapter, we will discuss the design and implementation of the priority assessment phase. The strategy underlying the updating of the priority assessment interval has a strong impact on the performance of the protocol. Several methods to handle the PAP will be presented and their implementation requirements will be discussed. A discussion of the robustness of the protocol will also be provided.

Chapter 4

Priority Assessment Phase Design And Implementation Considerations

As stated in the previous chapter, the protocol always grants the channel access right to the current highest priority class. However, several stations may belong to the latter class. Therefore, a mechanism to exclusively assign the channel access right to a unique station among all the stations currently holding a message with the highest priority is required. In a distributed real-time environment, the impossibility for a station to instantaneously know the present status of other stations makes the above goal hard to achieve. The necessity to satisfy the timing constraints imposed by the real-time application adds to the complexity of the problem. The support of real-time application requires an efficient implementation of the priority assessment mechanism. The overhead induced must be minimized in order to guarantee a good performance of the protocol.

The strategy underlying the updating of the priority assessment interval has a strong impact on the overall performance of the scheme. Different approaches can be used to achieve this purpose. The efficiency of these methods depends on their computational involvement and storage requirement. Some of the methods can achieve optimal control of the priority assessment interval. However, these methods usually involve a considerable amount of storage or computation. A trade-off between efficiency, computation involvement and storage requirement will help determine an adequate strategy for the scheme. The above issues and the proposed solutions are addressed in the following chapter.

In the first part of the chapter we introduce the design requirements of the PAP. The formal specification and the mathematical formulation of the PAP are provided. The second part of the chapter is dedicated to the implementation consideration of the PAP. A dynamic programming based approach as well as static and adaptive approaches are presented and the problem induced by their implementations are discussed. The last part of the chapter is devoted to discussing issues related to the protocol robustness and reliability. In that part, we will discuss the mechanism the protocol provides to handle faulty situations.

4.1 DESIGN REQUIREMENTS

In the proposed priority scheme, assessment of the highest priority class is required whenever a station attempts to revoke the right to transmit from the implicit virtual token holder.

The purpose of the PAP is to exclusively identify among all contending stations the station currently holding a message with the highest priority. It is the mechanism which transfers the virtual token from the current priority sub-ring to a sub-ring of higher priority. At the end of a successful transmission, all stations with priority higher than the priority of the implicit next virtual token holder, i.e., station *nv*, attempt to gain the right to transmit. Because of the broadcast nature of the transmission channel, both the implicit token holder and any revoking stations can detect the resulting collision, and these stations simultaneously enter the PAP. At the outset of the PAP, the contending stations form a logical *contention ring*. Low priority stations are successively dropped from the contention ring. As a result, the contention ring is reduced to the current highest priority sub-ring. If more than one station remains in the highest priority sub-ring, the token is assigned to the station in this sub-ring which is "closest" to the last virtual token holder.

Globally achieving fairness among all messages with the same priority becomes intractable in the absence of instantaneous information on the status of different stations. Selecting the station which is "closest" to the last virtual token holder as the next virtual token holder, forces the token to circulate in the sub-ring of highest priority in an attempt to achieve fairness at the station level.

The following section describes the mechanism which is used to select one among all the stations currently holding a message with the highest priority.

4.1.1 Formal Specification

Every station on a contention ring is uniquely identified by its *contention sequence number* (CSN). The assignment of these CSN's can be done either *statically* or *dynamically*. Static assignment uses the LID of a given station as its CSN.

Dynamic assignment of CSN's is based on the observation that only ready stations may compete for the channel access. Consequently, these stations are the only ones that must be assigned CSN's. A station is assigned a CSN when it successfully joins the network. The CSN assigned to the joining station is NA + 1, where NA represents the current number of ready stations. This value is carried in the NA field of the frame transmitted by the last ready station that successfully accessed the channel (Fig. 3.3). CSN's are revoked from stations when their state changes from ready to idle or when their failure is detected by the stations on the network. When a ready station successfully transmits the last message of its TQ, it sets the RP field of the transmitted frame to 0. All active stations detect the transition of the transmitting station from ready to idle, and decrement the current number of active stations by one. In addition, all stations with higher CSN's than the transmitting station decrement their own CSN's by one. The same updating procedure is used when the failure of a dormant station is detected by active stations on the network. Using the dynamic CSN assignment can reduce the time required to resolve PAP at the expense of the protocol complexity.

Based on the values of their current priorities and their CSN's, all contending stations generate a *contention parameter*. The objective of the PAP is reduced to exclusively identifying, among all contending stations, the station currently holding the highest contention parameter.

4.1.2 Mathematical Formulation

Formally, we need to develop a mapping that assigns a contention parameter to all contending stations based on their priority and their ordering on different priority subrings.

Let v be the CSN of the virtual token holder, the station that just successfully transmitted a message. Let n be the CSN of a given station on a given sub-ring. For simplicity, every station on the contention ring will be referred to by its CSN.

Let C be the size of the set of CSN's $\mathbf{C}$, $\mathbf{P}=\{0,1,\cdots,m-1\}$ be the set of priority classes, and $\mathbf{I}$ be a subset of integers, respectively. Notice that in the static assignment scheme, the set of CSN's is $\mathbf{C}=\{0,1,2,\cdots,N-1\}$, where N is the maximum number of stations in the network. In the dynamic assignment scheme, $\mathbf{C}$ is reduced to $\{0,1,2,\cdots,NA-1\}$, where NA is the current number of ready stations. The mapping function f_v is defined as

$$f_v : \mathbf{C}\times\mathbf{P}\rightarrow\mathbf{I}$$

The set of all possible contention parameters is referred to as *priority assessment interval* (PAI). The priority of the contending stations ranges between NP(nv), the current network priority, and the allowed highest priority $m-1$. Therefore only contention parameters generated for priorities higher than the current network priority will be used during the PAP.

Define *d(n,v)* be the *distance* or the number of arcs between station n and station v, the virtual token holder. Station n_1 is closer to v than station n_2 if and only if $d(n_1,v)<d(n_2,v)$. Formally, we have

$$d(n,v)=(C-v+n) \bmod C$$

For a contending station n with priority $NP(nv) \le p = p(n) \le m-1$, its contention parameter is $f_v(n,p)$. In order to minimize the expected length of the PAP, the PAI must be "hole-less". In other words, any number of the spanned interval must be a potential contention parameter. Thus, the function f_v must be a one-to-one mapping which preserves the ascendance property of the priority function. In addition, the assignment of the contention parameters must enforce the fairness requirement among stations with the same priority. As specified earlier, fairness is achieved at the station level based on the position of the station in the contention ring. The properties the mapping f_v must satisfy are summarized below.

- **Uniqueness**

  P1: $f_v(n_1,\mathrm{p}) \neq f_v(n_2,\mathrm{p})$, if $n_1 \neq n_2$.

  P2: $f_v(n,p_1) \neq f_v(n,p_2)$, if $p_1 \neq p_2$

- **Fairness**

  P3: If $d(n_1,v)<d(n_2,v)$, then $f_v(n_1,p)>f_v(n_2,p)$.

- **Priority ascendance**

  P4: $f_v(n,p_1)>f_v(n,p_2)$, if $p_1>p_2$.

  P5: $f_v(n_i,p_i)>f_v(n_j,p_j)$, if $p_i>p_j$.

- **Contiguity**

  P6: $|\ f_v(n_1,\mathrm{p}) - f_v(n_2,\mathrm{p})| = 1$, if $d(n_1,n_2)=1$.

  P7: $Min_{n_i}\{f_v(\mathrm{p},n_i)\} - Max_{n_j}\{(f_v(\mathrm{p}-1,n_j)\} = 1$.

Theorem 4.1

Let n be the CSN of a given station and p its priority.

Consider the following mapping f_v defined as follows :

$$f_v(n,p)=C(p-1)+(C-d(n,v)) \mathit{mod}\ C$$

This definition of f_v satisfies all of the above properties.

Proof

First, we prove the following: if $d(n_1,\nu)=d(n_2,\nu)$ *then* $n_1=n_2$

Assuming $d(n_1,\nu)=d(n_2,\nu)$, we can write

$(C-\nu+n_1)\ \mathit{mod}\ C=(C-\nu+n_2)\ \mathit{mod}\ C \Rightarrow$

$C-\nu-n_1=C-\nu+n_2+k\times C$, for some integer k, $\Rightarrow$

$n_1=n_2+k\times C$

However, $0\le n_1\le C-1 \Rightarrow k=0 \Rightarrow n_1=n_2$. **QED**

P1:

Assume that $f_\nu(n_1,p)=f_\nu(n_2,p) \Rightarrow$

$(C-d(n_1,\nu))\ \mathit{mod}\ C=(C-d(n_2,\nu))\ \mathit{mod}\ C \Rightarrow$

$d(n_1,\nu)=d(n_2,\nu)+k\times C$

However, $0\le d(n,\nu)<C \Rightarrow k=0 \Rightarrow d(n_1,\nu)=d(n_2,\nu) \Rightarrow n_1=n_2$. **QED**

P2:

Assume that $f_\nu(n,p_1)=f_\nu(n,p_2) \Rightarrow$

$C\times(p_1-1)+((C-d(n,\nu))\ \mathit{mod}\ C)=C\times(p_2-1)+((C-d(n,\nu))\ \mathit{mod}\ C)$;

Therefore, $p_1=p_2$. **QED**

P3:

Assume that $d(n_1,\nu) < d(n_2,\nu)$, we can write

$C - d(n_1,\nu) > C - d(n_2,\nu) \Rightarrow$

$(C - d(n_1,\nu))$ *mod* $C > (C - d(n_2,\nu))$ *mod* $C \Rightarrow$

$f_\nu(n_1,p) > f_\nu(n_2,p)$. **QED**

P4:

It is clear that if $p_1 > p_2$ then $C \times (p_1 - 1) > C \times (p_2 - 1) \Rightarrow$

$f_\nu(n,p_1) > f_\nu(n,p_2)$. **QED**

P5:

Assume that $p_i > p_j$ we can write

$p_i = p_j + k$, for some integer $k \geq 1$;

Therefore, $f_\nu(n_i,p_i) = C \times (p_j + k + 1) + ((C - d(n_i,\nu))$ *mod* $C)$, can be written as

$C \times (p_j - 1) + k \times C + ((C - d(n_i,\nu))$ *mod* $C)$, however, since $k \geq 1$, we can write

$k \times C + ((C - d(n_i,\nu))$ *mod* $C) > (C - d(n_j,C))$ *mod* $C = f_\nu(n_j,p_j)$. **QED**

P6:

We will assume without loss of generality that $f_\nu(n_1,p) > f_\nu(n_2,p)$;

We have $|\ (f_\nu(n_1,p) - f_\nu(n_2,p))| = ((C - d(n_1,\nu))$ *mod* $C) - ((C - d(n_2,\nu))$ *mod* $C)$;

The above equation reduces to

$((d(n_2,\nu) - d(n_1,\nu))$ *mod* $C)$, which can be further written as

$(((C - \nu + n_2)$ *mod* $C) - ((C - \nu + n_1)$ *mod* $C))$ *mod* C, which simplifies to

$(n_2 - n_1)$ *mod* $C = (C - (n_2 - n_1)$ *mod* $C = d(n_1,n_2)$.

Therefore, if $d(n_1,n_2) = 1$ then $|f_\nu(n_1,p) - f_\nu(n_2,p)| = 1$. **QED**

P7:

It is clear that for a given priority p, $Min_{n_i}\{f_\nu(n_i,p)\}$ and $Max_{n_j}\{f_\nu(n_j,p)\}$ are achieved for $n_i = \nu$ and $n_j = \nu + 1$, respectively.

Therefore, $Min_{n_i}\{f_\nu(n_i,p)\} = C \times (p-1) + ((C - d(\nu,\nu) \ mod\ C) = C \times (p-1)$

Similarly, $Max_{n_j}\{f_\nu(n_j,p-1)\} = C \times (p-2) + ((C - d(nu,\nu+1) \ mod\ C),$

The above equation reduces to $C \times (p-1) - 1 = Min_{n_i} - 1$.**QED**

This completes the proof of the theorem. □

Notice that the way the mapping is defined does not favor the token holder against the other contending stations. It does not allow the current token holder to send again if there is another contending station that has a message with the same highest priority. In the dynamic CSN assignment scheme, all joining stations use the value NA of the last successfully transmitted frame to compute their contention parameters. In order to enforcing uniqueness of the contention parameters among joining stations, each joining station has to append its unique LID to the computed contention parameter. Note that this concatenation does not introduce additional complexity in resolving the PAP.

4.1.2.1 Example

Consider the case where the number of active stations NA is 8 and the number of priority classes m is 3. Let station 2 be the current virtual token holder. Figure 4.1 shows the values of the contention parameters obtained for each class of priority and for each station. The values obtained show that for a given priority, p, station 3 holds the highest contention parameter among all the stations, station 4 the next highest, and so on. We can also note that the mapping preserves the increasing property of the priority function; higher priorities result in higher contention parameters.

Station LID	0	1	2	3	4	5	6	7
Distance	6	7	0	1	2	3	4	5

Figure 4.1-(a). Distance: d(n,v)

Station LID	0	1	2	3	4	5	6	7
C - Distance	2	1	0	7	6	5	4	3

Figure 4.1-(b). (C - d(n,v)) mod C

Station LID	0	1	2	3	4	5	6	7
p=1	2	1	0	7	6	5	4	3
p=2	10	9	8	15	14	13	12	11
p=3	18	17	16	23	22	21	20	19

Figure 4.1-(c). Contention parameters

Figure 4.1 Contention Parameters Computation

4.2 IMPLEMENTATION CONSIDERATION

Central to the PAP is a search procedure which successively examines a set of windows within the PAI until the highest contention parameter is determined. Starting with the initial priority assessment interval, successively smaller intervals are eventually generated according to a globally known search procedure. The formal description of the search procedure is shown in Figure 4.2. The procedure requires two arguments, the lower bound and upper bound of the PAI. Notice that the variables "low" and "up" represent the lower bound and the upper bound of the currently searched interval.

```
Procedure Search(min,max)
     begin
        done := false;
        low  := min
        up   := max;
        While ( (up-low) > 0 and ( not done) do
           begin
              bound := New_bound(low,up);
              If ( Channel Status = "collision" ) then
                 low := bound;
              elself (Channel Status = "idle") then
                 up := bound;
              else  (Channel Status = "transmission");
                 done := true;
           end
     end
```

Figure 4.2 Search Procedure

The protocol maintains a value of a lower bound such that only stations whose contention parameters are higher than the lower bound are allowed to contend for the channel. Note that each search step takes one slot time (channel round-trip propagation delay) to complete. Note also that because of the broadcast nature of the channel, all stations have a consistent view of the channel history.

When the initial value of the lower bound is chosen, one of three possible events may occur in the next slot:

(1) *Exactly one station responds to the open contention slot.* The attempt results in a successful transmission. The current highest priority station of the network is exclusively identified and the search terminates successfully.

(2) *More than one station attempts to transmit during the open slot.* The simultaneous attempts result in a collision. Thus the highest contention parameter is not yet determined. Further steps are required. The lower bound of the currently searched window is updated by a globally known procedure and the search resumes in the new window.

(3) *No station responds to the open contention slot* and the channel remains idle for one slot time. The highest contention parameter belongs to the lower window of the currently searched interval. The upper bound of the interval is updated to the equal the lower bound of the previously searched window and the search resumes in the new window.

The search procedure which identifies the station currently holding the highest contention parameter has a strong impact on the overall preformance of the priority scheme. The procedure relies on the strategy used to update the priority assessment interval. This strategy must be efficient in order to minimize the overhead incurred by the contention resolution. The overhead consists of the contention slots required to exclusively determine the highest contention parameter during the priority assessment phase. Reducing the number of steps required minimizes the expected length of the priority assessment interval. We define the length of a priority assessment interval as the number of contention slots needed before the station with the highest contention parameter is exclusively identified.

Minimizing the expected number of contention slots required to determine the highest contention parameter of the system depends on the strategy used to build successive priority assessment intervals. In other words, the search procedure should be able to determine the highest contention parameter in a minimum number of steps (or slots). In general, this number increases as the size of the initial priority assessment interval increases. Different methods could be used to achieve the above purpose. The basic features which characterize these methods, and on which depend the methods suitability for real-time application, include efficiency, computational involvement and storage requirement. These methods are classified as *dynamic*, *static* and *adaptive*. Dynamic methods are based on dynamic programming. Dynamic programming is a mathematical technique often useful for making a sequence of interrelated decisions. It provides a systematic procedure for determining the combination of decisions that maximize overall effectiveness. In essence, dynamic programming calculates the solution to all subproblems. The computation proceeds from the small subproblems to the larger subproblems. These methods usually produce an optimal solution. However, their computational involvement or storage requirement prohibits their use in a real-time environment. A atatic approach does not attempt to make use of any properties or characteristics of the input problem. The partitioning algorithm follows a predetermined pattern. Adaptive approaches try to strike a balance between efficiency, computation and storage requirements. In the remaining of this section we will briefly review the optimal solutions based on dynamic programming. We then provide a description of the different approaches we propose as alternatives to reduce the computational requirements of the optimal scheme. The performance analysis of the proposed schemes will help determine the adequate strategy for the scheme.

4.2.1 Optimal Approach

A central feature in the development of an optimal control policy of the PAP is the assumption that the PAP intervals are independent. Therefore, minimizing the length of individual PAI's tends to minimize the overall performance of the multiaccess protocol. Consequently, the optimal control of the length of the PAP can be achieved by choosing an optimum sequence of PAI. This approach was adopted in the optimal window protocol proposed by Wah and Juang [WhJu83,WaJu84]. The method, based on dynamic programming, provides a lower bound in terms of the number of contention slots required. In this method, the optimal control of the PAI is achieved by dynamically computing the boundary value of the next interval which reduces the number of future contention slots, in case the current contention slot results in a collision.

The expected number of contention slots can be computed by observing that a successful transmission will not require additional slots. However, if the channel remains idle or a collision occurs during the current contention slot, additional contention slots will be required in the updated PAI before the highest contention parameter is determined. Consequently, the *ith* value, L_i, of the PAI which minimizes the total number of contention slots can be obtained by solving the following recurrence equation:

$$\begin{aligned} M_v(low,up) = \min_{low<L_i<up} \{ & 1 + 0 \times S(L_i,low,up) \\ & + M_v(L_i,up) \times C(L_i,low,up) \\ & + M_v(low,L_i) \times I(L_i,low,up) \}; \end{aligned}$$

where

- $M_v(low,up)$ represents the minimum expected number of contention slots required to identify the highest contention parameter given that a collision occurred in the current PAI.

- $S(L_i, low, up)$ represents the probability of success in the next contention slot, when the next PAI used is $[L_i, up]$.
- $C(L_i, low, up)$ represents the probability of collision in the next contention slot, when the next PAI used is $[L_i, up]$.
- $I(L_i, low, up)$ represents the probability of channel remaining idle in the next contention slot, when the interval $[L_i, up]$ is used.

The dynamic programming approach discussed above provides a lower bound on the expected length of the PAP. However, its computational complexity is very high and cannot be afforded in the real time environment. An obvious alternative to avoid the computation is to store the different values generated by the search tree. The appropriate values will then be retrieved when needed. This alternative will certainly help reduce the computational complexity, but the storage requirement involved may be prohibitive. We propose a dynamic search scheme based on *optimal binary search* [AhHU76]. This dynamic scheme, also based on dynamic programming, requires less computation than the previous scheme and preserve to a high degree the "optimality" property of the dynamic window protocol.

4.2.1.1 Optimal binary search

The problem of achieving optimal control can be abstracted as efficiently processing a sequence of search instructions to determine the highest contention parameter in the current PAI. Given a subset $C=\{c_1,c_2,....,c_n\}$ of the set of all possible contention parameters, we need to design a data structure that allows the efficient processing of a search sequence with the smallest expected number of comparisons. Assume that, in addition to the subset C, we are given the probabilities $p_i, 1 \le i \le n$, that a given station generates a contention parameter of value $c_i, 0 \le i \le n$. A data structure that is suited for processing the search sequence is the optimal binary search tree.

```
Procedure Optree
    begin
      n:=1;
      for i:= NPL(nv) to m do
        for s:= 0 to NA-1 do
          begin
            c[n]:=$f_v$ (s,i);
            p[n]:=$p_{si}$;
            n:=n+1;
          end
      n:=n-1;weight[0,0]:=0;cost[0,0]:=0;
      for i:= 1 to n do
        begin
          weight[i,i]:=0;
          cost[i,i]:=0;
          r[i-1,i]:=i;
        end
      for m:= 1 to n do
        for i:= 0 to n-m do
          begin
            j:=i+m;
            weight[i,j]:=weight[i,j-1]+p[j];
            (* Let m be the value of k, r[i,j-1]≤k≤r[i+1,j], for
            which cost[i,k-1]+cost[k,j] is minimum *)
            cost[i,j]:=weight[i,j]+cost[i,m-1]+cost[m,j];
            r[i,j]:=m;
            root[i,j]:=$c_m$;
          end
    end
```

Figure 4.3 Optimal Search Tree

A closer look at the problem of achieving optimal control of the PAP reveals that the minimization of the expected number of contention slots does not only depend on the probability of success, but also on the choice of subsequent intervals. Let us assume the searched element c is the label $l(v)$ of some vertex v of the binary searched tree. The number of vertices visited before we reach the element c is one more than the depth* of the vertex v. Thus, the cost of the binary search search tree can be defined as:

$$Cost = \sum_{i=1}^{i=n} p_i \times (\, depth(a_i) + 1)$$

Consequently, optimal control of the PAI can be achieved by designing a binary search tree for C which minimizes the above cost.

The basic idea underlying the optimal binary search tree construction consists in determining the element pertaining to the root first. Once the above element is identified, the next step consists in recursively building the left and right subtrees of the binary search tree. The procedure is described in Figure 4.3. The algorithm requires $O(n^2)$ time to construct the optimal binary search tree [Knut73].

Once we have a minimum-cost binary search tree, we use the search method described in Figure 4.2 to determine the highest contention parameter. In the search procedure, the value returned by the function New_bound(i,j) is the root value of the optimal binary subtree build using the elements c_i, c_{i+1} , $\cdots$, c_j. This value is contained in the variable root[i,j]. Note that, since the elements searched frequently are more likely to appear at the root of an optimal binary subtree, the optimal tree need not be traversed entirely.

Dynamic programming based approaches provide a lower bound on the expected length of the PAP. However, these methods require either extensive computation or

*We define the depth of a vertex v in a tree as the length of the path from the root to v.

extensive storage. In addition, both methods assume that the probability p_{ij}, $0 \le i \le NA-1, 1 \le j \le m$, that station i generates a contention parameter when its priority is j are known. In some cases, the evaluation of these probabilities may present some difficulties.

The limitations of mathematical programming methods with respect to problem complexity were recognized from the very beginning, but the intent was to acquire a better understanding of how these optimal methods would compare against static and heuristic algorithms. In the next sections, we present alternative schemes which are less computationally involved. The efficiency of these methods will be compared with the results obtained for the optimal scheme.

4.2.2 Static Approach

A general static but less computationally involved method to solve the search problem is to use a dichotomizing based strategy. Given the value of the lower bound and the value of the upper bound, the next priority assessment interval can be obtained by considering the middle value of the currently searched interval.

The implementation of the dichotomizing based approach does not present any difficulty. The method involves only a shift operation and does not require any additional storage capacity. The computational complexity of the search procedure based on the dichotomizing approach is acceptable. By repeatedly splitting the searched interval in half we need never offer more than $\lceil \log(n+1) \rceil$, where n is the size of the initial PAI, contention slots to determine the highest contention parameter of the current priority assessment interval.

The dichotomizing based approach is simple to implement. However, the strategy is static and does not take into consideration the recent past information of the channel behavior. This information can be efficiently used to predict the immediate future behavior of the channel.

In the following section, we discuss adaptive methods which could be used in attempt to strike a balance between the optimal computationally involved methods and the static dichotomizing based approach.

4.2.3 Adaptive Approaches

As stated previously, a dynamic programming based approach requires time and space complexity proportional to n^2, where n is the size of the searched interval. Therefore, this scheme becomes impractical to use when n is large.

A closer look at the optimal search procedure reveals that the severe cost imposed by the dynamic programming approach is mostly due to the number of recursion steps required before the highest contention parameter is determined. This observation suggests the search of other alternatives to either eliminate or reduce the number of recursion steps. The elimination of recursion can be achieved by choosing the next boundary value of the current PAI as the value which maximizes the probability of success in the next contention slot. This value can be easily derived by setting the first derivative of the probability of success in the next contention slot to zero. The solution of the resulting equation determines the boundary value of the next PAI. Although not much computation is involved, this method only achieves a local optimization.

Recursion reduction can be achieved by combining dynamic programming approach and dichotomizing based approach. Since the size of the currently searched interval affects the number of recursion steps required, a satisfactory approach to reduce the number of recursion steps would consider the process of determining the highest contention parameter as a two step process. The first step consists of assessing the priority of the channel. This could be achieved by using a dichotomizing based approach. The second step consists of determining exclusively the station with the highest contention parameter currently holding a message of the assessed priority. A dynamic programming based approach could then be used to identify this station. Note that after the assessment of the channel priority, the size of the searched interval is reduced to the current number of active stations in the network.

The approximation methods discussed above offer an alternative to reduce the time complexity of the dynamic based schemes, preserving to a certain degree the optimality property of the dynamic programming scheme. However, the methods do not take advantage of the past information gathered from the channel.

The use at a given stage of the search of available observations from previous stages can provide a basis for a computationally efficient search. We introduce a useful model which can be used to include previously collected information into the current search. The basic idea behind the model is to compute the next boundary value of the currently searched interval, [Low, Up], as the average of all previous boundary values within that interval that lead to a successful transmission. Formally, $L_i(t)$ can be evaluated using the following equation:

$$L_i(t) = \frac{\sum_{k=1}^{k=q} \phi_k L_{t-k}}{\sum_{k=1}^{k=q} \phi_k}$$

where

$$\phi_k = \begin{cases} 1 & \text{if } Low \le L_{t-k} \le Up \\ 0 & 0 \; otherwise \end{cases}$$

A formal quantitative comparison of the three methods described above is rather difficult to carry out. One can obtain general analytical results for specific network configurations. However, a detailed analysis of each method involves too many parameters to be carried out in a tractable fashion. Therefore, one must rely on simulations to assess the performance of each method. The comparative study is the object of the next chapter. The remainder of this chapter is dedicated to discussing issues related to the protocol reliability and robustness and the mechanism the protocol provides to handle faulty situations.

4.3 RELIABILITY AND FAULT MANAGEMENT

In our investigation of the basic requirements that must be satisfied by any channel access mechanism intended to support communication in a distributed real-time environment, we emphasized the importance of the protocol robustness and reliability. In the following section, we will discuss the mechanism the protocol provides to handle fault recovery.

Due to spatial separation, stations cannot be guaranteed to have a common perception of the system state at any instant. Therefore, when a station receives a frame, it may infer that some station transmitted the frame, and therefore, that all active stations heard the transmission. However, the station does not make any inference about the validity or the nonvalidity of the frames other stations heard. In our discussion, we assume that stations may interfere with each others' transmissions, but cannot predictably alter the contents of transmitted frames. The latter observation restricts the discussion of the fault recovery mechanism to handling only those faults caused by station failures or communication errors. The failures include:

- Station failure,
- NPL inconsistency,
- Station out of synchronization.

In the remainder of the discussion, we first describe briefly the way the protocols handles intermittent stations failures. We then focus on the problem of mutual consistency which arises because of the existence of several copies of the NPL. In the final section, we discuss the way the protocol handles faults caused by stations which lose synchronization during a PAP.

4.3.1 Station Failure

The proposed protocol is entirely distributed and does not rely on a specific control message circulating among stations, such as a token. Since the token passing operation is implicit, the overall robustness of the proposed protocol is improved over the token passing bus and the token ring protocols. The proposed protocol does not require mechanisms to ensure recovery from a token loss, a token duplication, or a circulating busy token. Furthermore, the protocol does not rely on any knowledge of a particular network configuration such as the physical position of each station on the network. Therefore, the failure of a station has little impact on the normal behavior of the protocol. As described in the formal definition of the protocol, a failing station remains dormant until its failure is noticed by active stations in the network. The dormant station then becomes dead. Notice that a station's failure remains unknown to other stations in the network until the failing station becomes the next virtual token holder. Therefore, any station's failure which occurred while the station was idle remains totally transparent to other stations in the network. Upon its recovery, the station becomes joining. The behavior of a joining station was fully described in the formal definition of the protocol. It has been shown that the joining operation can easily be achieved without disturbing the functioning of the network.

4.3.2 Network Priority List Inconsistency

In spite of its robustness against intermittent station failures, the reliability of the proposed protocol still depends on the ability of the stations to accurately maintain its NPL. The NPL entry of the transmitting frame must be updated at the end of the frame transmission. The protocol makes use of the broadcast nature of the bus to allow active stations in the network to dynamically update their NPL's. However, unreliable storage media as well as transmission errors can cause inconsistency to occur between different copies of the NPL. The discrepancies between the actual values of the protocol global parameters, namely the network priority and the virtual token holder, and the value of these parameters as stored by a given station may cause the faulty station to disturb the normal behavior of the protocol.

The technique used to ensure a robust and reliable functioning of the protocol is based on the redundancy of the information describing the global parameters of the protocol. These values are available in the NPL of each station and carried by the last successful transmission. The redundancy of this information allows the station to detect the inconsistency of its NPL and prevents it from altering the normal behavior of the protocol. Each time an active station detects a discrepancy between the value of the network priority carried by the successfully transmitted frame and the value contained in its NPL, the station discards the content of its NPL and becomes a joining station. Similarly, after receiving a frame, all active stations check the transmitted frame for eventual transmission errors, by verifying the validity of the accompanying frame check sequence (FCS). Consequently, in the absence of transmission errors, the resulting update sequence must be strictly identical for all stations, resulting in consistent copies of the NPL. However, if a given station detects the nonvalidity of the transmitted frame, and since the station cannot make any inference about what other stations have heard, the station assumes the inconsistency of its NPL and moves into the joining state. Notice that in both cases, the station behavior is passive and does not affect the protocol operations. In the case when the station is the next virtual token holder the channel may remain idle. After one slot time, active stations assume that the virtual token holder has failed and a new epoch is resumed with the next virtual token holder. Notice that when a station detects a transmission error, an error count is incremented and if a predetermined threshold is reached over a given time interval, an indication of the condition is reported. The report notification can either be local or remote to another node providing network management.

4.3.3 Station Out Of Synchronization

The redundancy of the global parameters adds to the robustness of the protocol. Nevertheless, situations may occur when a station loses synchronization during a priority assessment phase. The faulty station may disturb the regular behavior of the search process, since it proceeds within a different PAI than other non-faulty active contending stations.

Let l_f be the lower bound of the current PAI as computed by a faulty station s_f, and c_f the value of the contention parameter generated by this station. At a given stage of the PAP, six distinct cases can be identified. These case are listed below:

$$c_f < l < l_f \tag{1}$$

$$c_f < l_f < l \tag{2}$$

$$l < c_f < l_f \tag{3}$$

$$l < l_f < c_f \tag{4}$$

$$l_f < l < c_f \tag{5}$$

$$l_f < c_f < l \tag{6}$$

In the first three cases, station s_f does not affect the search process, since its contention parameter is lower than l_f. The faulty behavior of station s_f will remain transparent until the next successful transmission. The station then realizes the inconsistency of its local information and moves into the joining state. In the last three cases, the faulty station causes a collision to happen in the next slot. In the cases of (4) and (5), the PAP is legitimate since the station's contention parameter is higher than the lower bound of the PAI. In the last case, however, the PAP is not legitimate. At the outcome of the PAP, one of two possible events may occur.

- A unique station is selected and successfully transmits its frame. Depending on the identity of the transmitting station and the priority of the transmitted frame, the faulty behavior of the station may or may not remain transparent to other stations in the network. In all cases, however, a new epoch of the protocol normal behavior resumes at the end of the transmission.
- Either the out-of-synchronization station or the remaining active stations detect a collision while a leaf of their respective PAI tree is being searched. In either case, the station who detects that a leaf is being searched withdraws from the contention and refrains from transmitting. This will either allow the faulty station to transmit in the next slot or the PAP to continue properly. In all cases, however, all stations enter synchronously the next phase of the protocol at the end of the successful transmission.

As is apparent from the above discussion, the protocol provides an acceptable mechanism to handle faulty situations. In essence, we have proven that faulty situations are transient and do not affect markedly the protocol normal behavior. The redundancy of the protocol global information allows faulty stations to recover without causing the total failure of the protocol.

4.4 CONCLUSION

In this chapter, we described the mechanism the protocol uses to exclusively assign the channel access right to a unique channel among all contending stations. The search procedure which identifies this station has a strong impact on the overall performance of the protocol. We presented several approaches which could be used in the strategy underlying the updating of the priority assessment interval. We argued that the dynamic programming based approaches are optimal, but their computational involvement is prohibitive in a real-time environment. Static approaches are simple to implement but do not take advantage of the history of the channel which is available to all stations because of the broadcasting nature of the transmission media. We proposed an adaptive approach which attempts to strike a balance between the dynamic and static methods. A comparative study of these methods will help determine their efficiencies. This study is the topic of the next chapter.

Finally, we discussed the protocol reliability and robustness. We have shown that the proposed protocol, based on the concept of a virtual token, eliminates unnecessary token passing around idle stations. Without a physical token, the faulty situations created by the loss of token or duplication of token will never occur. Thus, the fault management scheme is rather simple compared with the standard token-passing scheme. Furthermore, we have shown that most of the faulty situations are transient in nature and do not cause the total failure of the protocol.

Chapter 5

Comparative Study and Sensitivity Analysis

The previous chapter concentrated primarily on discussing the priority assessment phase of the protocol. Three basic methods, dynamic, static and adaptive, which could be used to assess the network priority and grant the channel to a unique stations among all contending ones, have been described. As noted in the previous chapter, the PAP has a strong impact on the overall performance of the protocol. The objective of the following chapter is to test each strategy in some representative network situations and allow conclusions to be drawn about the effectiveness of each method on the protocol performance evaluated according to some specific efficiency criterion. A sensitivity analysis of the protocol to certain parameters of the system will also be conducted.

In the first part, we lay out a set of considerations and terms which must be considered in the design of the simulation model. An attempt will be made to determine reasonable values for each parameter of the simulation in order to capture the performance profile of the protocol. The second part of this chapter is aimed at developing a comparative analysis of the three PAI searching methods. The performance study will evaluate the effectiveness of the protocol under each method. A quantitative comparison will be established. In the third part, we provide results of simulations to compare the proposed scheme and the standard schemes, namely CSMA/CD and token passing bus. The last part of the simulation study deals with the sensitivity of the protocol to specific network parameters. An analysis of the effect on the protocol behavior of certain parameters, such as the network size and the traffic distribution among stations and priority classes, on the protocol behavior is provided.

5.1 SIMULATION MODEL AND PERFORMANCE PROFILE

A formal quantitative comparison of the three searching methods described in the previous chapter is rather difficult to carry out. One can obtain general analytical results for specific network configuration. However, a detailed analysis of the performance of each method involves too many parameters to be carried out in a tractable fashion. Therefore, one must rely on simulation. Simulation is the traditional technique for performance evaluation when detailed characterization of the system is needed. It provides an effective tool to analyze the dynamic behavior and produce the performance features of systems whose analytical modeling appears to be mathematically untractable.

The objective of the following section is to provide a description of a simulator which is used to investigate the performance of the protocol and verify fulfillment of the desired requirements under various system configurations.

5.1.1 Simulation Model

To investigate the performance profile of the proposed virtual token-passing protocol under different search schemes, a simulation model was developed. The simulation program was written in *CSIM* [Schw85] and developed on a VAX-8600/UNIX machine. CSIM is a process oriented simulation language based on the C programming language. Process oriented simulation is a convenient tool for developing simulation models for distributed systems. The primary unit of execution is a process. All currently active processes execute in a quasi-parallel fashion. This feature provides an effective and easy way to express the dynamic behavior of the system.

The input parameters of the simulation model specify the network configuration and the load characteristics. The network configuration parameters specify the number of clusters on the network, the size of each cluster, the channel bandwidth and the number of priority classes defined in the network. The load characteristics include the network load and the parameters controlling the distribution of the load among different classes of messages, for each cluster. Each class is characterized by the length, the priority and the deadline associated with its messages.

The model assumes that messages from class j arrive at node i in a *Poisson* stream at rate α_{ij} messages per seconds. Each message requires $\bar{x}_{ij}$ seconds of transmission time. The rate of arrival of messages differs from one class to another. However, nodes of the same cluster generate messages of different classes at the same rates. We define the network load as follows:

$$\rho = \sum_{i=0}^{N-1} \sum_{j=0}^{C-1} \alpha_{ij}\bar{x}_{ij} = \alpha\bar{x}$$

where N represents the total number of stations in the network and and C represents the number of message classes defined in the system. Therefore, when the system load, the message length and the load proportion of each class are specified, the message arrival rate per node for each class is decided by the above formula.

5.1.2 Performance Profile

The performance metrics of the system include:

- The *percentage of message loss*, for each class. This performance measure is defined as the ratio of the number of lost messages of a given class to the total number of message of the same class successfully transmitted.
- The *effective channel utilization*, defined as the ratio of the total time the channel was used for successful transmission to the total time units simulated.

- The *message delay* for each priority class, defined as the time elapsed between the instant the message is generated and the instant the message is received.
- The *average size* of the priority assessment phase expressed in terms of slots.

The quantitative variables mentioned above will go some distance toward accomplishing the objective of the simulation, in so far as they will provide a map of how each station in the network handles its messages. For real-time applications, the two first performance metrics are of major importance in the performance profile of the system. A protocol dedicated to handling real-time traffic should minimize the message loss and reduce the channel bandwidth wasted due to collisions. Both performance objectives are crucial in meeting the timing requirements of real-time messages.

5.2 COMPARATIVE STUDY

The previous chapter has concentrated primarily on describing mechanisms that can be used in assessing the priority of the channel. In this section, we focus on the impact of each method on the protocol performance. First, we introduce the mathematical model used to compute the probability $c_i(j)$, that station i generates a contention parameter of value j. We then describe the simulation experiments which were used to evaluate the performance of the protocol under the static scheme, the adaptive scheme and the dynamic scheme. The simulation results will be used to assess and compare the performance of the protocol under the different searching methods.

5.2.1 Mathematical Model

As stated in the previous chapter, the problem of achieving optimal control of the PAP can be abstracted as efficiently traversing a binary tree until the highest contention parameter is encountered. We argued that a suitable data structure to efficiently handle the traversal is the optimal binary search tree. In the following section, we describe the model that is used to determine the probabilities necessary to build the optimal tree.

Let λ_{ij} represent the message mean arrival rate of station-i with priority-j and μ_{ij} the corresponding service rate. Message interarrival times are assumed to be exponentially distributed. In the steady state, the network throughput is $\lambda = \sum_{i=0}^{N-1} \sum_{j=1}^{m-1} \lambda_{ij}$. Let μ_j be the mean service rate of class-j messages, which is a function of the frame size. A class-j message can be transmitted if no message with priority higher than j exists in the network. Thus, the effective mean service rate of class-j message is [Klei76]

$$\mu_j^e = \mu_j \prod_{k=j+1}^{m-1} (1-\rho_k)$$

where ρ_k is the traffic intensity of class-k messages and can be recursively defined as

$$\rho_k = \frac{\sum_{i=0}^{N-1} \lambda_{ik}}{\mu_k^e}$$

The effective service rate of class-j messages of station-i, μ_{ij}^e, and the corresponding traffic intensity, ρ_{ij}, can be respectively defined as

$$\mu_{ij}^e = \mu_j^e \frac{\lambda_{ij}}{\sum_{i=0}^{N-1} \lambda_{ij}} \quad \text{and} \quad \rho_{ij} = \frac{\lambda_{ij}}{\mu_{ij}^e}$$

In the dynamic search scheme of PAP, the probability $c_i(j)$ that the station-i generates a contention of value j must be known and can be defined as [JuWa84]

$$c_i(j) = \rho_{ij} \prod_{k=j+1}^{m-1} (1-\rho_{ik}) \quad \text{for } 0 \le i \le N-1 \text{ and } 1 \le j \le m-1.$$

5.2.2 Simulation Results

A series of simulation experiments were conducted to compare the performance of the protocol under the three different PAI searching methods described in the previous chapter. A description of these experiments and a discussion of their results is the object of the following section.

Although the proposed protocol can handle various faulty situations as well as the insertion and deletion of stations, the simulator assumes an error-free transmission medium and a fixed number of fully operational stations in the network. In a distributed real-time environment, the occurrence of transmission error and faulty stations is rare. Thus, the time spent in fault handling is rather small and is not simulated. The dynamics of collision handling are determined by a parameter called the slot time. The parameter describes two important aspects of collision handling. In essence the slot time represents:

- an upper bound on the acquisition time of the network,
- an upper bound on the length of a frame fragment generated by a collision.

In the first experiment, the comparative performance evaluations of the proposed virtual token passing protocol were made relative to a load description commonly found in industrial environment [Lela87]. The network load originates from workshops, telephones and factories. Five classes of messages are generated by the stations on the network. The network configuration and the message characteristics were chosen to represent an environment as close as possible to a real situation. The heterogeneity of the messages adds to the complexity of the simulation model but is intended as a concession to reality. The network configuration and the characteristics of each class of messages is summarized in Table 5.1. The simulated network was comprised of three clusters of six stations each. The first cluster simulated the traffic generated by the workshops, the second cluster the traffic generated by the telephones and the third cluster the traffic generated by the factories. The bandwidth of the channel was assumed to be 10 Mbits/sec.

Traffic	Frame length (bits)	Priority	Deadline (ms)	Average number of messages/sec/station		
				Workshops	Telephone	Factory
Files	16000	1	50	0.5	0	0.1
Transactions	1600	2	20	10	0	2
Telephones	2000	3	15	0	32	14.6
Sensors/Actuators	240	3	5	80	0	20
Alarms	240	4	0.9	1	0	0.25

Table 5.1 Workload Characteristics.

The value of the slot time was 0.020 *ms*. The simulation ran until 100,000 messages were generated. We started collecting data after 5000 messages were simulated. This was necessary for the simulation model to reach an equilibrium state where the fluctuations are reduced to less than 5%.

The results showing the delay-throughput characteristics of each class of messages for the static and dynamic searching methods are summarized in Figure 5.1. The results showing the average response time and the average number of steps required to assess the priority of the channel are summarized in Figure 5.2. The remaining results showing the percentage of late messages for are described in Table 5.2. These results are only reported for the static scheme since the results of the two other schemes were not significantly different.

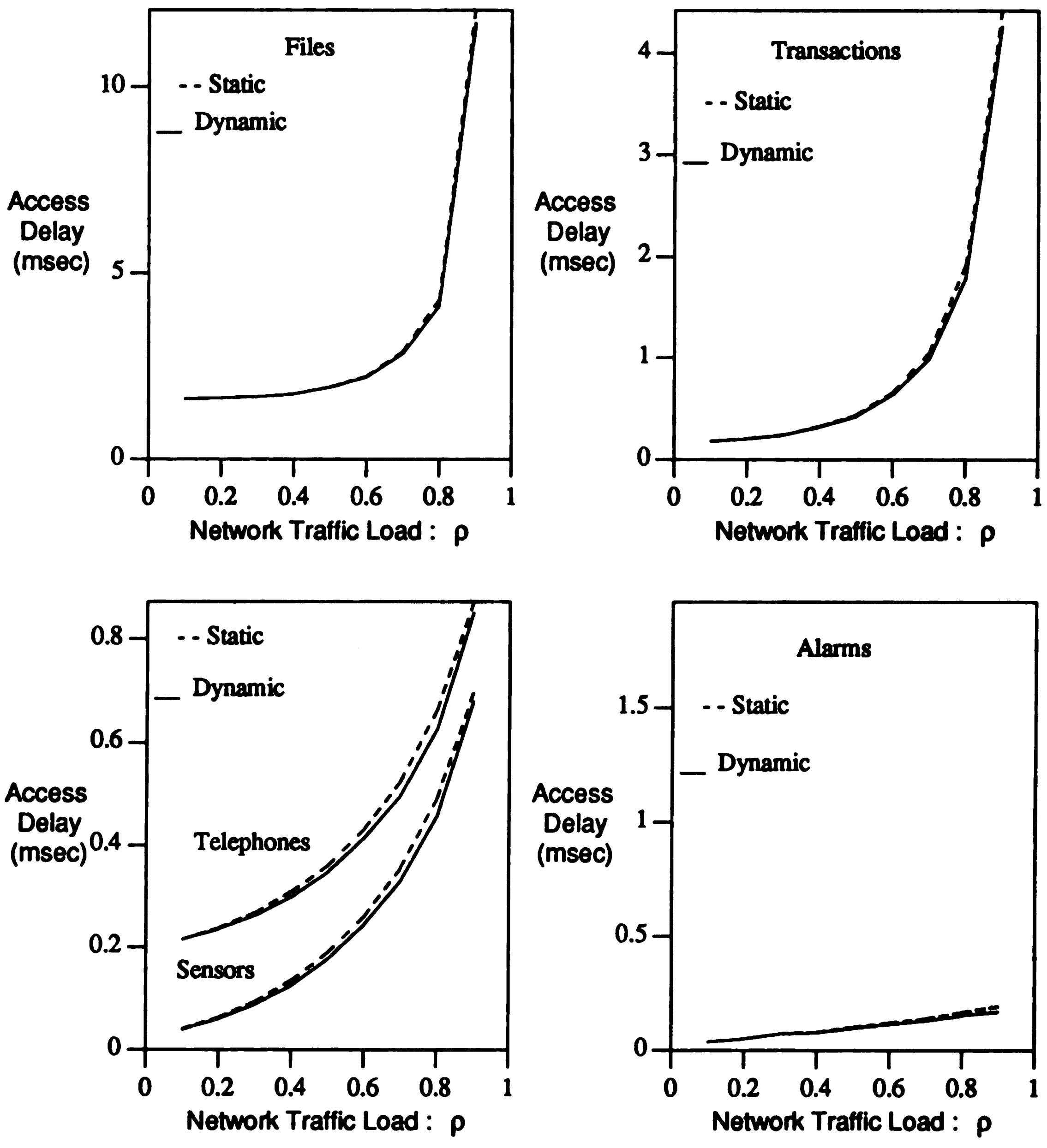

Figure 5.1 Protocol Throughput Delay Characteristics.

Network Offered Load	.1	.2	.3	.4	.5	.6	.7	.8	.9
Average Response Time	0.108	0.130	0.160	0.205	0.260	0.344	0.460	0.657	1.064
Average Number of Steps	3.310	3.331	3.385	3.370	3.360	3.360	3.343	3.325	3.312

(a)- Adaptive Approach

Network Offered Load	.1	.2	.3	.4	.5	.6	.7	.8	.9
Average Response Time	0.108	0.129	0.158	0.197	0.251	0.330	0.434	0.617	1.036
Average Number of Steps	2.092	2.136	2.370	2.286	2.389	2.499	2.407	2.476	2.865

(b)- Dynamic Approach

Network Offered Load	.1	.2	.3	.4	.5	.6	.7	.8	.9
Average Response Time	0.108	0.131	0.162	0.207	0.265	0.347	0.464	0.659	1.066
Average Number of Steps	3.316	3.333	3.389	3.371	3.361	3.363	3.347	3.328	3.317

(c)- Static Approach

Figure 5.2 Performance Profile of Different Searching Strategies

Based on the results of the first experiment, we observe the following:

- The static and adaptive approaches performed very closely. In most cases, the results obtained for different performance profile parameters were similar. This indicates that knowledge of the channel history did not produce a significant improvement over the static method in minimizing the number of steps required to determine the current highest contention parameter. This is due to the fact that in most cases, and because of the reservation mechanism, the PAI was "sparse". Therefore, the number of steps required was usually relatively small.
- The optimal binary search tree based approach produced better results than the adaptive and static schemes. However, the improvement was not significant enough to justify the cost of computation involved in the dynamic based approach.

Offered Load	.1	.2	.3	.4	.5	.6	.7	.8	.9
Workshop Traffic Loss	0.00005	0.00007	0.00008	0.00015	0.00059	0.0009	0.0069	0.067	0.210
Telephones Traffic Loss	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.008
Factory Traffic Loss	0.0	0.0	0.00001	0.00008	0.00004	0.00017	0.0002	0.0004	0.144

Table 5.2 Percentage of Late Messages

(Static Approach)

In order to further compare the impact of the PAI searching strategy on the overhead needed to implement the ideal scheme described in chapter 3, a second experiment was conducted. All transmitted frames have the same length which was equal to 1000 bits. The channel bandwidth was assumed to be 10 Mbits/sec, while the slot time was chosen to be equal to the time necessary to transfer 512 bits. In the first part of this experiment, the network was comprised of 15 stations and 3 different priority classes. The network load was equally shared among all the stations, i.e. $\lambda_i = \lambda / 15$ for $0 \le i \le 14$. For each station, the lowest priority class was responsible for 60 % of the generated traffic. The remaining two priority classes were each responsible for 20 % of the station traffic. More specifically, for all stations i of the network $\lambda_{i1}=0.6\lambda_i$, $\lambda_{i2}=\lambda_{i3}=0.2\lambda_i$.

	Static	Dynamic	
Load	Avg Nbr. of Steps	Avg Nbr. of Steps	Frequency of PAP
.1	2.160	1.909	0.002
.2	2.161	1.933	0.011
.3	2.208	1.971	0.025
.4	2.227	1.989	0.048
.5	2.251	2.034	0.077
.6	2.308	2.036	0.117
.7	2.328	2.094	0.163
.8	2.371	2.165	0.203
.9	2.381	2.202	0.204

Table 5.3 Effect of the System Load on the Performance of the PAP

Table 5.3 shows the average number of steps required to resolve the PAP for both dynamic and static search schemes as well as the occurrence frequency of the PAP. The minimum number of steps required is one (caused by the first collision to enter the PAP) and the maximum number of steps required is $\lceil \log_2 N(m-1) \rceil = \lceil \log_2(15\times3) \rceil = 6$. In this case also, the dynamic scheme performs better than the static scheme. However, the difference is still small. As the system load increases, the number of steps increases, again not significantly. The simulation results indicate that the priority assessment interval (PAI) was sparse and, in general, two to three steps are enough to exclusively determine the station with the highest contention parameter.

In the second part of this experiment, the system load is kept fixed with, $\lambda=0.5$, and the number of stations varied from 5 to 35. The results obtained are reported in Table 5.4.

	Static	Dynamic	
Nbr of Stations	Avg Nbr. of Steps	Avg Nbr. of steps	Frequency of PAP
5	1.985	1.816	0.062
10	2.151	1.931	0.068
15	2.252	2.034	0.070
20	2.277	2.021	0.072
25	2.304	2.009	0.072
30	2.352	2.143	0.072
35	2.388	2.096	0.072

Table 5.4 Effect of the Number of Stations on the PAP Performance

Again, the dynamic scheme performance metrics are slightly better than the static scheme. We also notice that as the number of stations increases, the length of the PAI increases, causing the average number of steps to increase. However, since the network load is fixed, the increase of the number of stations cause the decrease of each station contribution to the load. Consequently, the number of contending stations did not considerably increase as the size of the network increased. As a result, the increase of the average number of steps required to assess the priority of the network was still small, as the size of the network grew larger.

Both Table 5.3 and Table 5.4 also show the occurrence frequency of the PAP per successful message transmission. When the network load is light, the value of the occurrence frequency is very small because the channel is likely idle. Thus at load light situation, the network behaves like CSMA/CD. When the load becomes heavy, the PAP occurs, on the average, every five successful transmissions. However, since the average duration of the PAP is rather small compared to a successful transmission, the overhead induced by the PAP is not significant and does not affect the performance of the protocol.

Based on the above results, it is clear that a static binary search tree based approach is sufficient to achieve acceptable results with a minimum computational overhead. The static method will be adopted for further study of the protocol performance.

5.3 PERFORMANCE COMPARISON WITH STANDARD SCHEMES

In this section, we compare the performance profile of the proposed virtual token protocol and the IEEE standard schemes, namely CSMA/CD and token passing bus. The PAI searching method adopted in the virtual token protocol is static and uses the dichotomizing approach. The comparison of these schemes is based on the delay-throughput characteristics and the percentage of lost messages. The network parameters used for the simulation, and the profile of the different classes of messages were selected based on the values described in Table 5.1 [Lela87].

The technical characteristics used, such as frame format and inter-frame spacing, conform to the IEEE 802 standards [IEEE82, IEEE83]. The slot time was chosen to be equal to 0.020 *ms*.

The simulation was carried out for two different network sizes. In the first simulation experiment, the network was comprised of 52 stations. In the second experiment, the network included 102 stations. Figure 5.3 shows the delay-throughput characteristics of the traffic generated by the workshop and the percentage of lost messages, obtained for CSMA/CD, token passing bus and the virtual token protocol, when the number of stations in the network was 52. The results obtained for the telephone traffic and the factory traffic are summarized in Figure 5.4 and Figure 5.5 respectively. The performance profile of the proposed virtual token protocol, when the network is comprised of 102 stations are reported in Figure 5.6. The results include the average response time, the average number of steps required to resolve the collision, the average delay of the workshop traffic, the telephone traffic and the factory traffic and the percentage of late messages, for different loads of the network.

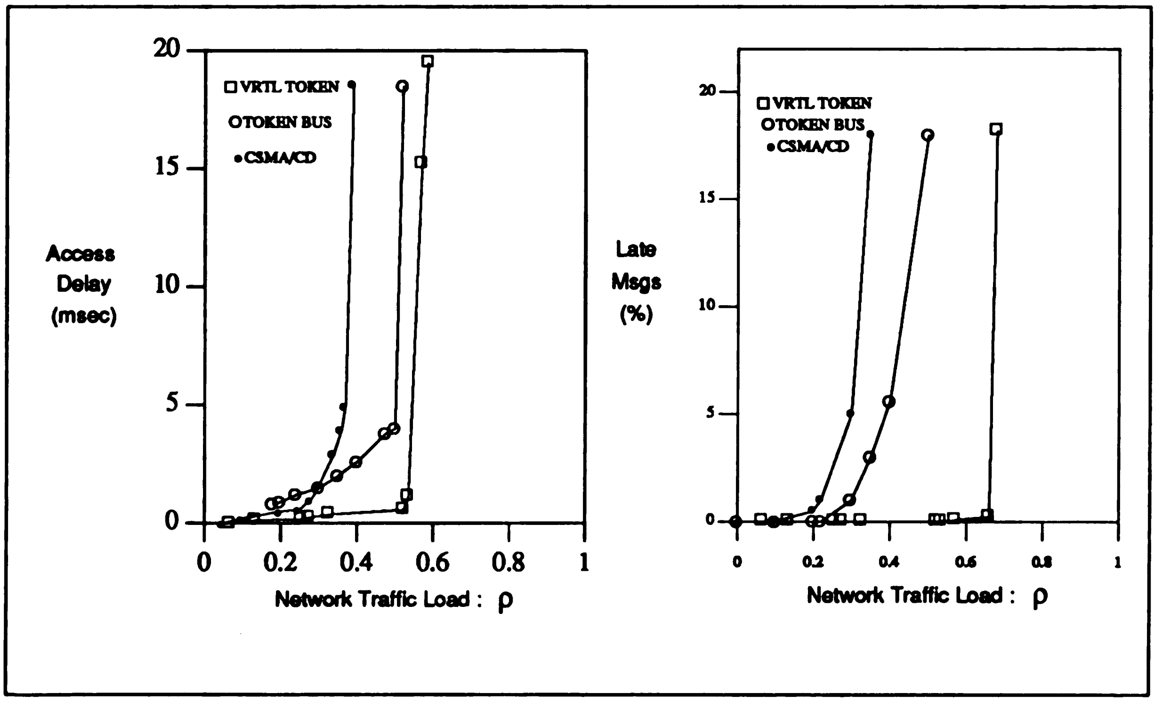

Figure 5.3 Performance Profile of the Workshop Traffic

The results show that the virtual token passing scheme fares better than the two standard schemes, for the three types of traffic. Furthermore, the results show that in light load situations, the performance of the proposed protocol is comparable to the performance of the CSMA scheme. However, in a heavy load situation and because of its efficient priority mechanism, the performance of the virtual token passing protocol is superior to the performance of both standard schemes. This is to be expected since, CSMA/CD does not support any priority functions, while in the token passing bus scheme, the token passes through a large number of nodes which are either idle or holding messages for which the token-rotation timers have expired. In the proposed scheme,

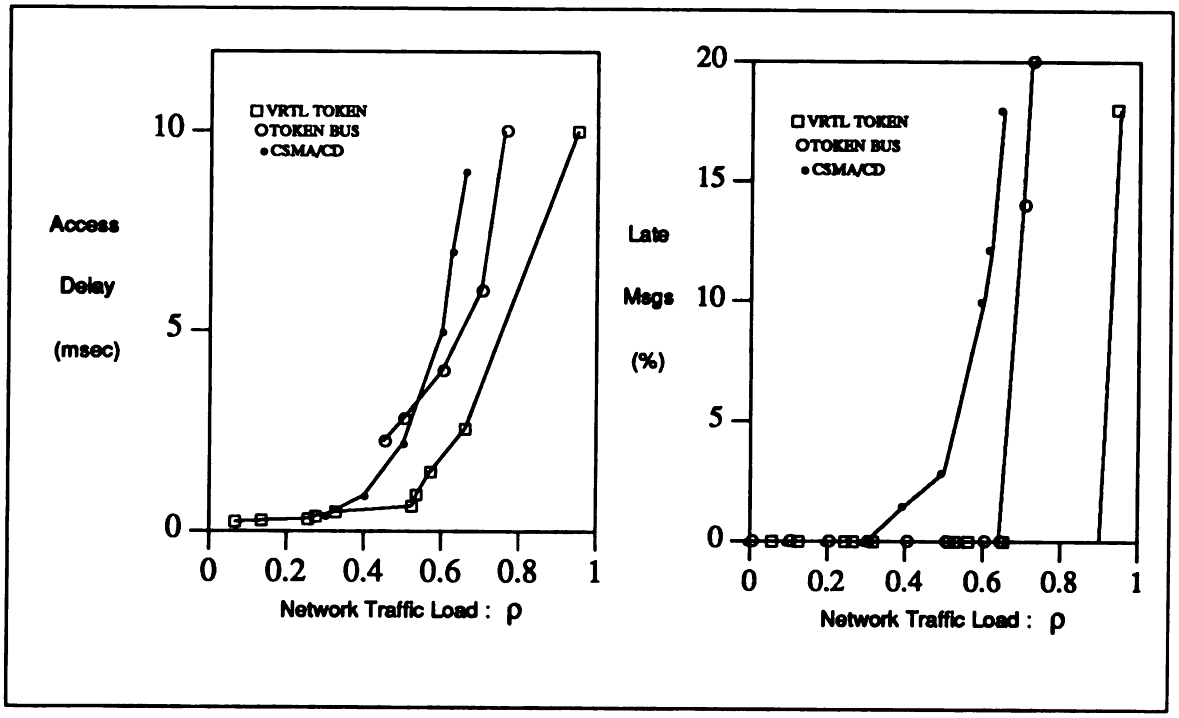

Figure 5.4 Performance Profile of the Telephone Traffic

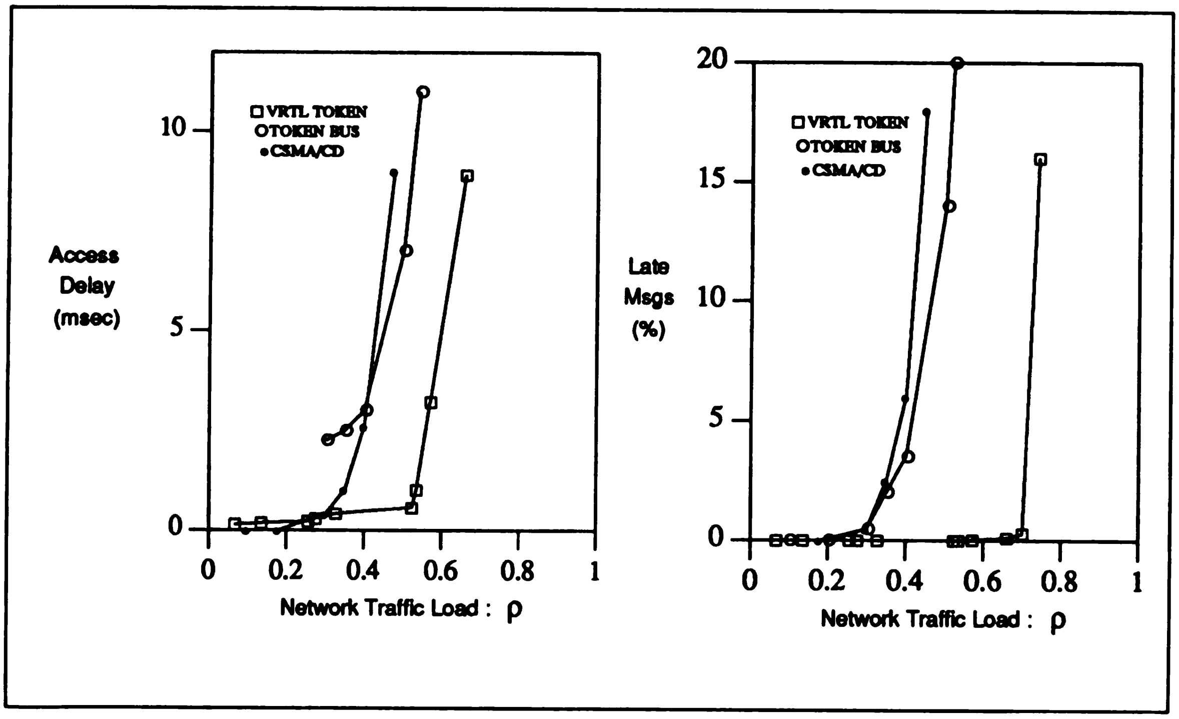

Figure 5.5 Performance Profile of the Factory Traffic

since the channel is always granted to the highest priority class, lower priority classes suffer longer delays. However, since the deadlines of these classes is large compared to

Offered Load	.1	.2	.3	.4	.5	.6	.7	.8	.9
Average Response Time (msec)	0.128	0.160	0.205	0.284	0.405	0.630	1.132	8.564	466.571
Average Number of Steps	3.755	3.788	3.803	3.918	3.968	4.113	4.238	4.308	5.176
Workshop Delay (msec)	0.084	0.116	0.163	0.245	0.379	0.630	1.221	13.01	793.517
Telephone Delay (msec)	0.236	0.267	0.309	0.380	0.481	0.659	0.953	1.553	3.090
Factory Delay (msec)	0.143	0.276	0.222	0.297	0.402	0.605	1.066	3.465	51.138
File transfer Loss (%)	0.0	0.0	0.0	0.0	0.0	0.0	0.025	0.77	0.91
Transactions Loss (%)	0.0	0.0	0.0	0.0	0.0	0.00053	0.0225	0.39	0.97
Phone messages Loss (%)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.002	0.01
Sensors Loss (%)	0.0	0.0	0.0	0.0	0.0	0.0025	0.008	0.033	0.16
Alarms Loss (%)	0.0	0.0	0.0	0.0	0.02	0.020	0.019	0.034	0.006
Workshop Loss (%)	0.0	0.0	0.0	0.0	0.0	0.0025	0.010	0.076	0.26
Telephone Loss (%)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.003	0.01
Factory Loss (%)	0.00	0.0	0.0	0.0	0.0	0.0	0.006	0.045	0.16

Network Size : 102

Figure 5.7 Protocol Performance Profile

higher priority classes, the percentage of late messages was kept low. Furthermore, the losses, whenever they occurred, affected the lower priority classes. This is acceptable since the loss of lower priority classes is less crucial than the loss of messages of higher priority.

5.4 SENSITIVITY ANALYSIS AND EVALUATION CRITERIA FULFILLMENT

In the previous section we performed a comparative study of the search methods which could be used to exclusively identify the station which has the highest contention parameter during the PAP. The main conclusion of the above study has been to observe the effective performance of the dichotomizing based search approach. Because of the severe limitations of the dynamic programming based approach with respect to time complexity and memory requirements, and the non-significant improvement of the adaptive scheme, dichotomizing based approach appears to be the most appealing method to implement an efficient and cost-affordable search procedure.

As stated previously, the size of the network did not have a major impact on the overhead incurred by the PAP. As the size of the network grew larger, the average number of steps required to resolve the PAP increased. However, the resulting increase in the message delay was still relatively small compared to the message transmission time. In the following section, we further investigate the performance of the protocol for different network configurations, using the dichotomizing based search. The main purpose of the experiments outlined below is to determine the sensitivity of the protocol to the network load distribution, among stations and priority classes, and the protocol fulfillment of the evaluation criteria of a priority scheme.

5.4.1 Effect of Load Distribution

In order to study the effect of the load distribution of the performance profile of the protocol, four cases of simulation studies were conducted. Each case was characterized by the type of traffic distribution among the stations and the type of traffic distribution among the priority classes of messages within the same station. Two types of traffic distribution, *symmetric* and *asymmetric* were considered. In the symmetric case, the mean message arrival rate at each station is assumed to be the same. In the asymmetric case, however, the stations were divided into clusters. Each cluster was responsible for a certain percentage of the traffic offered to the network. Stations of the same cluster have the same mean message arrival. Similarly, in the symmetric priority class case, the station load is equally distributed among the priority classes. In the asymmetric case, however, each priority class is responsible for a certain portion of the station traffic.

In the first experiment, the network comprised 36 stations. The results showing the average response time for each priority class of messages and for each type of load distribution are shown in Figure 5.7. The results prove that the load distribution among stations did not have a significant impact on the delay characteristics of the priority classes. In the other hand, the stations' load distribution among priority classes had a slight impact on the average response time and on the average number of steps required in the assessment of the highest contention parameter. The relative variation in the average message delay, however, did not exceed 0.014 %. The same experiment was conducted with different network sizes, namely 12 and 72 stations. Similar results were observed. This proves the protocol's capability to handle different load distributions with no excessive overhead.

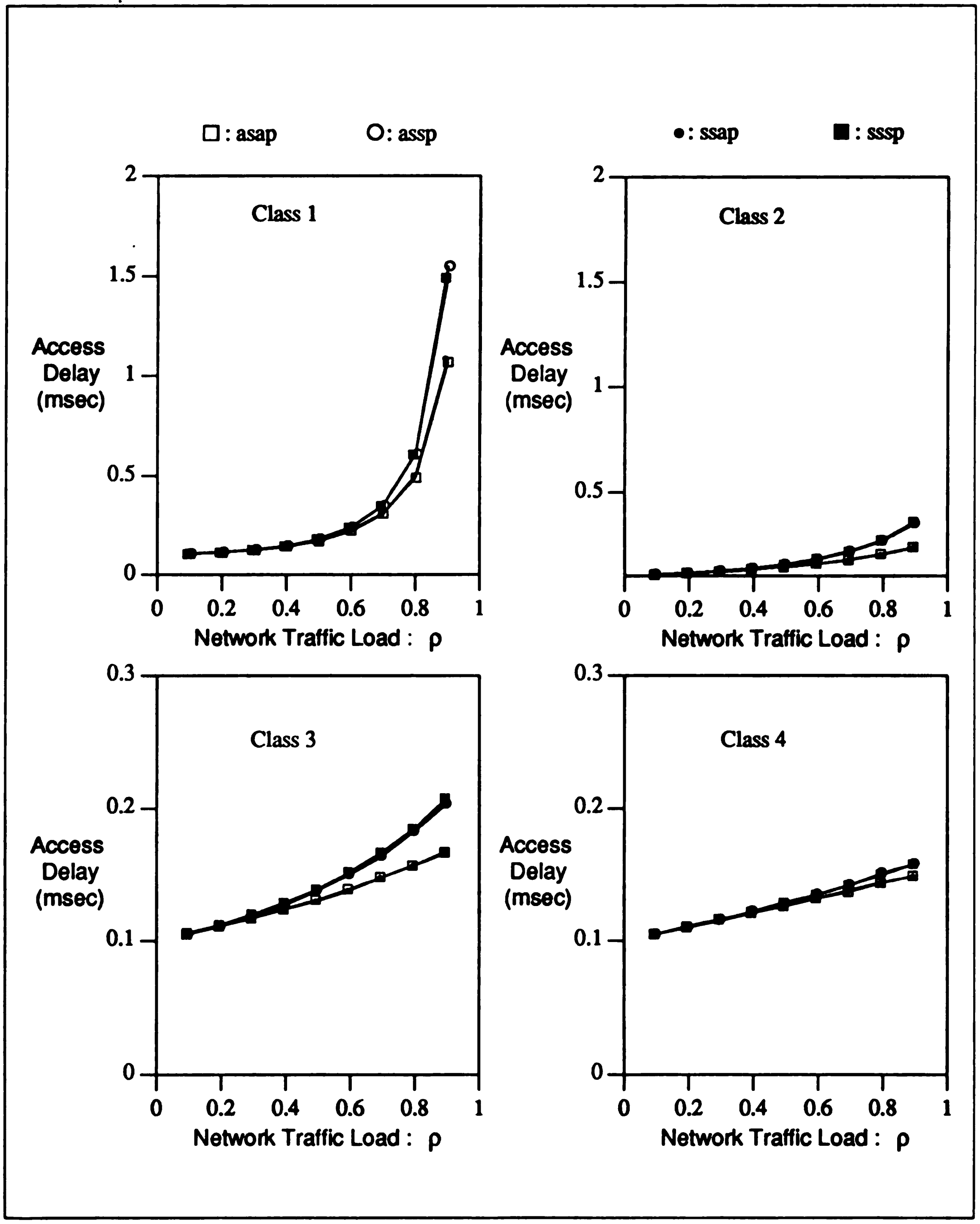

Figure 5.7 Effect of the Load and Priority Distribution on the Delay.

5.4.2 Protocol Adherence to Priority Scheme Requirements

The remainder of this section is dedicated to study the adherence of the proposed virtual token scheme to the evaluation criteria of a priority scheme, stated earlier [RoTo81].

In the description of the protocol, we have argued that achieving global fairness requires an excessive amount of overhead. In our approach, we guarantee fairness among stations with the same priority by granting the virtual token to the closest station to the

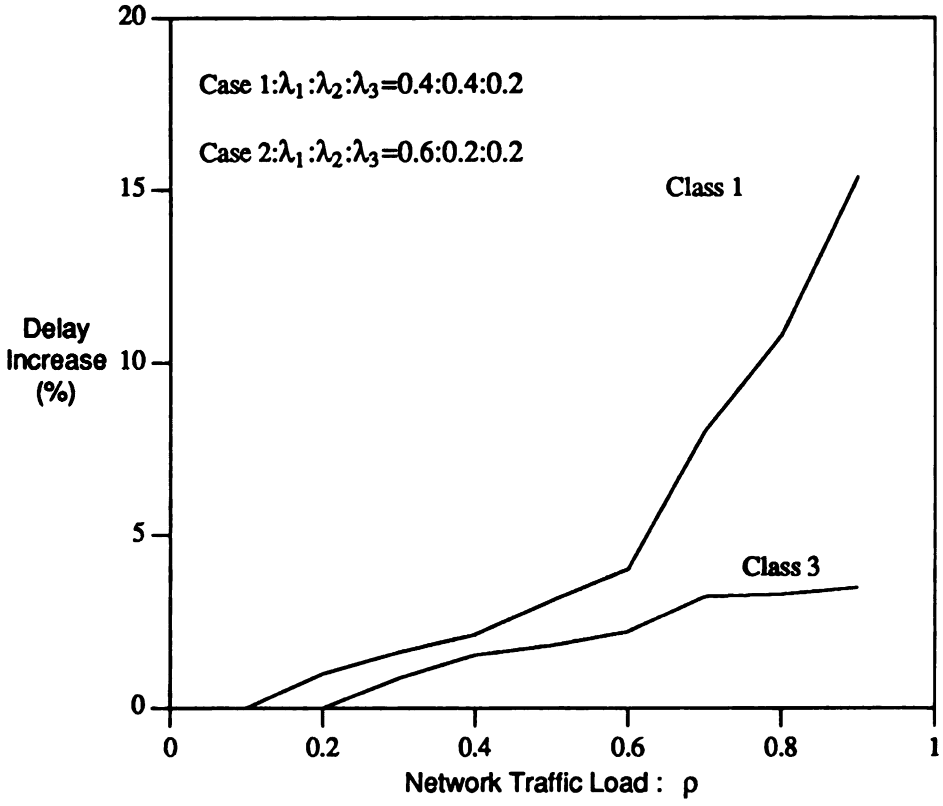

Figure 5.8 Protocol Hierarchical Performance

last virtual token holder along the logical ring. Issues related to the protocol robustness and reliability have already been discussed in the previous chapter. As stated earlier, the protocol does not rely on a specific control message circulating among messages. This characteristic improves the overall robustness of the proposed protocol over the token passing bus and the token ring protocols. In addition, we argued that faulty situations are transient and do not drastically affect the behavior of the protocol. We have also proven that the overhead required to grant the channel to the current highest priority class is small and does not significantly affect the system performance. Furthermore, we have shown that in the proposed scheme, idle channels are skipped. This considerably reduces the overhead required by the token passing scheme to regulate the access to the channel. The remainder of the discussion will concentrate on the last criterion for a priority scheme, namely the hierarchical independence of performance. In order to study hierarchical performance of the protocol, we designed an experiment to simulate a network of 36 stations and 3 priority classes. In the first case of the simulation, the two lowest priority classes were responsible for 80 % of the network traffic. In the second case, the load of the second highest priority class was doubled. Figure 5.8 shows the relative increase of the message delay for the network lower and higher priority classes. The results show that the increase of the second highest priority class load did not cause a significant increase of the message delay of higher priority classes. The increase did not exceed 4 % in a heavy load situation. In the other hand, the lowest priority class was penalized. An increase of the access delay of this class was observed. This demonstrates the hierarchical independence of priority class performance requirement can be achieved with a minimum overhead.

5.5 CONCLUSION

In this chapter, we described a set of simulation experiments which aimed at comparing the different strategies the protocol can use in order to exclusively determine the station with the highest contention parameter during the PAP. The main outcome of the comparative study was that the static approach was sufficient to achieve acceptable results with a minimum computational overhead. The simulation results have also demonstrated that the overhead spent in implementing the protocol is relatively small.

Based on the static approach, a performance comparison of the proposed protocol with the standard schemes, CSMA/CD and token passing bus, was provided. The results prove that in a light load situation, the protocol performed very closely to the CSMA/CD scheme. In a heavy load situation, and because of its efficient priority mechanism, the proposed protocol provides significant improvement in performance compared to the performance of the standard schemes.

In the last part of the chapter, we described a simulation experiment to study the sensitivity of the protocol to the network load distribution and its adherence to the priority function evaluation criteria. The results show that the increase of lower priority class messages does not have a significant impact on the performance of higher priority class messages.

Chapter 6

Analytical Modeling of Multi-access Protocols

Various protocols for channel access arbitration in bus networks have been proposed in the literature. In this chapter we present a general model which can be used to analyze and compare the steady state behavior of these protocols, in a unified manner. The global behavior of a multiaccess protocol for a bus network is characterized by a transmission phase, a resolution phase, and an idle phase. Steady state equations for a general service distribution are derived for each phase using the method of supplementary variables. The resulting model is then employed to determine the steady state behavior of two classes of networks: those with a finite number of stations and those with a number of stations large enough to be considered infinite. Numerical examples are then given for two specific service distributions to illustrate the effect of specific parameters on the performance of a multiaccess protocol, and the application of the model to the virtual token passing-based protocol is provided.

6.1 INTRODUCTION

The increasing use of data communication as part of computer systems over the past several years has spawned a variety of network architectures to support requirements for distributed processing . As stated in a previous chapter, network architectures identify the physical and logical elements of a communication system and specify the permitted interconnection and interactions among the elements of the system. Typically a network architecture specifies peer protocols and indicates the functional relationship between layers. Protocols establish logical communication paths between communicating entities. The design of distributed multiaccess protocols involves several factors which influence overall system behavior, performance and efficiency. Some factors are absolute requirements to ensure the correctness of the protocol; other factors deal with the intelligence of the decisions made by the protocol and the overhead involved. The latter factors have a major impact on the network performance.

In this Chapter we are concerned with a *multiaccess* environment in which a single channel provides the only means of communication among geographically distributed stations. All stations can monitor and detect the activity of the channel. If more than one station attempts to access the channel simultaneously, a *collision* occurs resulting in the loss of information. A *distributed multiaccess* protocol grants channel access to a single station among all contending stations. A typical example of distributed multiaccess protocols is the class of multiaccess protocols developed for a shared-bus local area network in which the shared bus is the only means of communication among the stations [ChFr79,KuSY84,ZnNi87]. These protocols include the CSMA/CD protocol, token-passing protocol, and urn protocol [Lam83].

Common to most of the distributed multiaccess protocols is a *resolution phase* which determines among all competing stations the next station to be allowed access to the channel. During the resolution phase, the channel is not allocated to any of the waiting stations, resulting in a waste of resource capacity. Therefore, it becomes of interest to evaluate the overhead induced by the resolution phase upon the utilization of the channel.

The analysis of distributed multiaccess protocols is complicated by the number of factors which influence the aggregate behavior of the system. Two distinct approaches have been used to model the performance of multiaccess protocols. In the first approach, embedded Markov processes and renewal theory have been used to carry out an exact steady-state queueing analysis [KlTo75,RoTo81,FrBi80,Toba80,IIYO80,ToHu80]. The second approach is based on approximation [JuWa84]. The main purpose of this chapter is to abstract a general mathematical model that can represent various multiaccess protocols for bus networks in a unified manner. The proposed model is based on the supplementary variables approach. This method is well known in principle [Coxd54,RaJa69]. We show that this method is a tractable approach to analyze the behavior of distributed multiaccess protocols. We first provide a formal description of the analytical model. We then detail the analysis of the model and derive the steady state equations of the system. These equations are then applied to solve for networks with finite numbers of stations and networks with a number of stations large enough to be considered infinite. We conclude the chapter by applying the model to evaluate the delay-throughput performance of the virtual token protocol described in the previous chapters [ZnNi87].

6.2 ANALYTICAL MODEL DESCRIPTION

In the first part of this section, we introduce a state transition table to qualitatively describe the general behavior of distributed multiaccess protocols . The second part describes mathematically the model used to analyze the behavior of the distributed multiaccess protocols.

6.2.1 Qualitative Description of the Model

As previously discussed, only a single station can successfully transmit a message at any given time. Conflicts arise when more than one station attempts to access the channel simultaneously. In order to resolve these conflicts, a multiaccess protocol that allocates the channel to one among the competing stations is required. At the occurrence of a collision, no actual service is performed by the channel and the distributed multiaccess protocol enters the resolution phase. At the completion of this phase, a unique station, among all contending stations, wins the right to access the channel.

We assume that the current state of the channel can be sensed and recognized by all stations. Three channel states , *transmission state, resolution state* and *idle state* can be identified. In the first state, the channel is allocated to a station for which it is performing service. The second state indicates that a conflict is being resolved among contending stations attempting to access the bus. Notice that when the channel is in the resolution state, no message is actually being transmitted. We assume that the resolution phase of the protocol may require several steps before a unique station is granted access to the channel. Both the total number of steps and the time required by each step are not known a priori. In other words, the number of these steps and the amount of information exchanged during the assessment of the contention is an inherent characteristic of the protocol, and thus may vary from one protocol to another. In most protocols, the contention step time represents the end to end propagation delay required to ensure that all stations in the network are able to detect a collision or an idle state of the channel. The third state, idle state, refers to an empty system where no station in the network is currently holding a message for transmission.

A *state transition table* (STT) notation is used to describe the behavior of the protocol. The STT permits a graphical view of the behavioral dynamics of the shared channel in a distributed environment and also provides the details of the underlying stochastic process. The table will be useful in the establishment of the differential difference equations of the stochastic evolution of the underlying process. Every state of the channel is described by a state transition table (Figures 6.1-(a),(b) and (c)). The "event" entry of the table describes the type of event that may occur while the system is in the current state and the rate at which the event occurs. The "effect" entry determines the changes induced by the occurring event. Transition among states is dictated by a set of conditions described in the "condition" entry of the state transition table. Notice that in some cases the system moves into a transient state before reaching the next state. The detailed description of these tables will be provided in the next section.

Current state	[N(t),S(t),Y(t)]=[0,0,0] : Idle	
Event	ϕ	Arrival($\lambda(n)$)
Effect		N(t)=1
Condition		
Transition state		[1,1,0†]
Next state	Idle	Transmission
Transition index	0	1

Figure 6.1-(a) Idle state transition table.

6.2.2 Stochastic Formulation of the Model

Analysis and performance evaluation of multiaccess protocols is usually difficult owing to the multiplicity of conditionality and correlation that exists among the various parameters involved in the protocol. These parameters have a strong impact on the system behavior and thus their abstraction in the analytical model is important in order to correctly predict the behavior of the protocol. However, the complexity of the relationships among these parameters results in a mathematically untractable model yielding a complex set of equations often hard to solve. Therefore, a set of simplifying assumptions is to be made for the model to be tractable. We assume the following

current state	$[N(t),S(t),Y(t)]=[n,1,y_1(t)]$: Transmission				
Event	End of transmission($\eta_1(y_1(t))$)			Arrival($\lambda(n)$)	ϕ
Effect	N(t)=n-1			N(t)=n+1	
Condition	N(t)>0 & No Conflict (1-$\beta(n)$)	N(t)>0 & Conflict $\beta(n)$	N(t)=0		
Transient state	$[n,1,0_1^+]$	$[n,2,0_2^+]$	[0,0,0]		
Next state	Transmission	Resolution	Idle	Transmission	Transmission
Transition index	2	3	4	5	6

Figure 6.1-(b) Transmission state transition table.

- Stations have buffer space for only one message,
- Newly generated packets form an independent Poisson process with mean arrival rate $\lambda(n)$, where n is the total number of stations currently in the system. A station is either in the system, if it is currently holding a message for transmission, or outside the system, if its transmission buffer is empty. In the following study the infinite source model as well as the finite source model will be considered. In the first model, the number of stations in the network is large enough to be assumed infinite. In the second model, the total number of stations in the network is fixed.
- The transmission time of a message is drawn from a general distribution function $B_1(x)$ with a density function $b_1(x)$. We denote by $\eta_1(x)dx+o(dx)$ the first-order conditional probability that a transmission completes in the interval $(x,x+dx)$ if it has been in service for time x, so that

current state	$[N(t),S(t),Y(t)]=[n,2,y_2(t)]$: Resolution			
Event	End of current resolution step($\eta_2(y_2(t))$)		Arrival($\lambda(0)$)	ϕ
Effect			N(t)=n+1	
Condition	Resolved($\alpha(n)$)	Not resolved(1-$\alpha(n)$)		
Transient state	$[n,1,0_1^+]$	$[n,2,0_2^+]$		
Next state	Transmission	Resolution	Resolution	Resolution
Transition index	7	8	9	10

Figure 6.1-(c) Resolution state transition table.

$$b_1(x) = \eta_1(x)\, e^{-\int_0^x \eta_1(u)du}$$

- Similarly, we denote the density function of the resolution step time by $b_2(x)$ and the first-order conditional probability that the current resolution step completes in the interval $(x,x+dx)$ if it did not complete before x by $\eta_2(x)+o(dx)$. Evidently,

$$b_2(x) = \eta_2(x)\, e^{-\int_0^x \eta_2(u)du}$$

- Let n represent the current number of stations contending for the channel. We denote by $\beta(n)$ the probability that a collision occurs at the end of transmission of the current message and by $\alpha(n)$ the probability that the resolution phase ends at the completion of the current resolution step. More specifically, $\alpha(n)$ represents the probability that the current resolution step is the last step required before a unique station is granted the channel access. Notice that the values of $\alpha(n)$ and $\beta(n)$ are protocol dependent.

6.2.3 Method of Analysis

Before we outline the method of analysis which we use in the study, we briefly introduce the important terms used in the description of the model. A *stochastic process* is a process in which the state of the system changes with a parameter, usually time, in a probabilistic manner. An interesting class of stochastic processes is the Markov process. A stochastic process is said to be a Markov process if the present state of the system is sufficient to predict the future without the knowledge of the past history of the system. Stochastic processes which do not exhibit the Markovian property are termed non-Markovian. The analysis of non-Markovian processes frequently results in a mathematically untractable model. However, it is often possible to treat stochastic process of non-Markovian nature by reducing it to a Markov process. Several techniques for studying non-Markovian processes are available in the literature. The most commonly used techniques include the *method of stages* [Klei74], *Lindley's integral equation* [Lindi52], and the *method of supplementary variables* [Kend53, Cox55]. Beyond these approaches there exist other approaches to non-Markovian processes, among which are the *random walk*, the combinatorial approaches [Taka67] and the method of *Green's function* [Kril65].

The method of stages is based on the notion of *decomposing* the service time distribution into a collection of structured exponential distributions. Therefore, so long as the the interarrival time and the service time probability distribution functions have Laplace transforms that are rational, the stage method applies. The method gives a procedure for carrying out a solution but does not show the solution as an explicit expression. Therefore, the properties of the solution cannot be studied for a class of systems. The embedded Markov chain-based technique involves extracting a discrete-time Markov chain embedded in the continuous-time process. The set of points, usually referred to as regeneration points, verify the Markovian property. The main difficulty presented by this method resides in identifying those instants of state transitions during which the process behaves like a Markov chain. The approach based on Lindley's integral equation is suitable for queueing systems with general interarrival time distributions and general service time distributions. However, the method cannot be specialized easily to particular systems. The fundamental idea behind the method of supplementary variables is to include in the state vector of the stochastic process, the complete information which summarizes the past history relevant to the future process behavior. In general, so long as the system state description is not complex, the method provides an efficient tool to study the performance of several classes of systems. In this study, the approach adopted is based on the supplementary variable method as discussed below.

6.2.3.1 System state vector

Consider the stochastic process $\{N(t), t>0\}$, where $N(t)$ denotes the number of stations requesting access to the channel. This process is not Markovian because the future behavior of the system cannot be predicted from the knowledge of the present number of stations requesting access to the channel. In order to make the system Markovian, we need to incorporate the missing information by adding "supplementary variables" to the state description, namely the state of the channel, $S(t)$, and the elapsed time that the channel has been in the current state. This time is denote by, $y_1(t)$, when the channel is currently servicing a message, and denoted by, $y_2(t)$, when the channel is in the resolution state. Formally, the evolution of the stochastic process describing the dynamic behavior of the channel can be fully described by the state vector $[N(t), S(t), Y(t)]$, where

- $\{N(t), t>0\}$ represents the current number of stations requesting access to the channel, including the station currently transmitting,
- $\{S(t), t>0\}$ denotes the state of the channel. The idle state, transmission state and resolution state are denoted by 0, 1 and 2, respectively,

- $\{Y(t), t > 0\}$ represents either the amount of transmission time already received by the station currently transmitting, if the channel is being used, or the elapsed time in the current step of the contention resolution phase.

The resulting process $\{[N(t),S(t),Y(t)], t > 0\}$ is a Markov process and is an appropriate state vector for the system, since it completely summarizes all past history relevant to the future system behavior. Following is a description of the evolution of this process through time.

When in state 0, the system remains in this state (transition 0 of the STT) until an arrival occurs at rate $\lambda(n)$. The occurrence of an arrival increases $N(t)$ by one and causes the system to move into the transmission state(transition 1). When the system is in the transmission state, transmission is being attempted. In this state, arrivals may occur as a homogeneous Poisson process of rate $\lambda(n)$ (transition 5). We assume that new arrivals sense the channel busy and will not interfere with the ongoing transmission, although the model can be slightly modified to accommodate for service interruption. The end of transmission (service) denotes a departure from the system. At the instant of a departure, $N(t)$ is decreased by one and $Y(t)$ becomes zero. If the waiting queue is empty, the system moves into the idle state (transition 4), and remains in this state until the occurrence of an arrival. However, if the queue of transmitting stations is not empty, one of two possible evolution will take place. Either no contention resolution is required; the system remains in the transmission state (transition 2), and the next station accesses the channel, or a contention resolution is required in which case the system moves into the resolution state (transition 3) until the conflict is resolved. During the resolution phase of the protocol, no service is delivered but new arrivals may occur at the homogeneous rate $\lambda(n)$ (transition 9). At the end of the resolution phase, the system moves into transmission state (transition 7). However, if at least another step is required in the resolution of the access conflict the system remains in state 2, (transition 8), and a new resolution step is performed. In all STT's, ϕ denotes the non-occurrence of any event, when the system is in the state specified in the STT.

6.3 Resolution Of The Model

We begin by expressing the Chapman-Kolmogorov dynamics of the system. Let $Z_t = \{[N(t),S(t),Y(t)], t \varepsilon R^+\}$ be the state of the system at time t. Its state space is $I \times [0,1,2] \times R^+$, where I is the set of positive integers representing the number of stations in the network and R^+ is the set of positive real numbers.

We assume that the distribution of Z_t converges to a stationary law on $I \times [0,1,2] \times R^+$. The necessary and sufficient conditions to achieve this state will be stated later. We define

$$p(n,s,y_s,t) = \lim_{dy_s \to 0} \frac{1}{dy_s} Pr[N(t)=n, S(t)=s, y_s \le Y_s \le y_s + dy_s];$$

$$s=1,\ 2;\ n \ge 2 \text{ and } y_s > 0.$$

as the joint probability that at time t, there are n stations requesting access to the channel including the station currently transmitting, the channel is currently in state 1 and the elapsed transmission time of the message being transmitted lies between y_s and $y_s + dy_s$.

To obtain the differential difference equations connecting these probabilities, we relate the state of the system at time time $t+dt$ to the state of the system at time t. Assuming $S(t)=1$, $n \ge 2$, and $y_1 > 0$ we obtain

$$p[n,1,y_1+dt,t+dt] = p[n-1,1,y_1,t]\ \lambda(n-1)\ dt$$
$$+ p[n,1,y_1,t](1-(\lambda(n)+\eta_1(y_1))dt) + o(dt);$$

where $o(dt)$ means that $o(dt)/dt$ tends to zero as dt tends to zero. Rearranging the right-hand side of the above equation we obtain

$$p[n,1,y_1+dt,t+dt] - p[n,1,y_1,t] = p[n-1,1,y_1,t]\ \lambda\ (n-1)\ dt$$
$$- p[n,1,y_1,t]\ (\lambda(n)+\eta_1(y_1))\ dt$$

By adding and subtracting the quantity p[n, 1, y_1 +dt, t] in the left hand side of the equation and by dividing by *dt*, we get

$$\frac{p[n,1,y_1+dt,t+dt]-p[n,1,y_1+dt,t]}{dt}+\frac{p[n,1,y_1+dt,t]-p[n,1,y_1,t]}{dt}=$$

$$p[n-1,1,y_1,t]\lambda(n-1)-p[n,1,y_1,t](\lambda(n)+\eta_1(y_1))+\frac{o(dt)}{dt}.$$

Taking the limit as $dt\to 0$, the above equation reduces to

$$\frac{\partial}{\partial t}p[n,1,y_1,t]+\frac{\partial}{\partial y_1}p[n,1,y_1,t]=\lambda(n-1)\,p[n-1,1,y_1,t]-$$

$$(\lambda(n)+\eta_1(y_1))\,p[n,1,y_1,t];$$

However at the steady state, the probabilities do not depend on time. Therefore, $\frac{\partial}{\partial t}$p[n,1,$y_1$,t]=0. Consequently the above equation reduces to

$$\frac{\partial}{\partial y_1}p(n,1,y_1)=\lambda(n-1)\,p(n-1,1,y_1)-(\lambda(n)+\eta_1(y_1))\,p(n,1,y_1),$$

where $p(n,1,y)=\lim_{t\to\infty}p[n,1,y,t]$.

The same derivation is applied when n=1 and results in

$$\frac{\partial}{\partial y_1}p(1,1,y_1)=-(\lambda(1)+\eta_1(y_1))\,p(1,1,y_1).$$

At the completion of a busy period, the system moves into the idle state. The departure from the idle state can only be due to the arrival of a newly generated message . Therefore, the resulting differential difference equation is

$$p[0,0,0,t+dt]=(1-\lambda(0)dt)\,p[0,0,0,t+dt]+$$

$$\int_0^\infty p[1,1,y_1,t]\eta_1(y_1)dy_1+o(dt).$$

At the steady state, the above equation reduces to

$$\lambda(0)\, p(0,0,0) = \int_0^{\infty} p(1,1,y_1)\, \eta_1(y_1) dy_1$$

Similarly, we can derive the differential difference equations connecting state probabilities when the system is the resolution state. The resulting equation is

$$\frac{\partial}{\partial y_2} p(n,2,y_2) = \lambda(n-1)\, p(n-1,2,y_2) + (\lambda(n)+\eta_2(y_2))\, p(n,2,y_2).$$

At the end of a contention resolution phase, or when no contention is required at the end of the current transmission, the distribution function $p[n,s,Y(t) \le y_s,t]$ is reduced to a mass function $p[n,s,Y(t)=0_s^+,t]dt$. Consequently we can write, assuming $n \ge 2$

$$p[n,1,0_1^+,t+dt]dt = \alpha(n) \int_0^{\infty} p[n,2,y_2,t]\eta_2(y_2)\, dy_2\, dt$$

$$+ (1-\beta(n)) \int_0^{\infty} p[n+1,1,y_1,t]\eta_1(y_1) dy_u) dt + o(dt)$$

Dividing both sides by dt and making $dt \to 0$ we get

$$p[n,1,0,t] = \alpha(n) \int_0^{\infty} p[n,2,y_2,t]\eta_2(y_2) dy_2 + (1-\beta(n)) \int_0^{\infty} p[n+1,1,y_1,t]\eta_1(y_1) dy_1;$$

At the steady state, the above equation reduces to

$$p(n,1,0) = \alpha(n) \int_0^{\infty} p(n,2,y_2)\eta_2(y_2) dy_2 +$$

$$(1-\beta(n)) \int_0^{\infty} p(n+1,1,y_1)\eta_1(y_1) dy_1;$$

When the number of stations requesting access to the channel is reduced to one, we obtain

$$p(1,1,0) = (1-\beta(1)) \int_0^\infty p(2,1,y_1)\eta_1(y_1)dy_1 + \lambda(0)p(0,0,0);$$

Similarly, we can derive the boundary conditions for the resolution state. The steady state equation obtained reduces to

$$p(n,2,0) = (1-\alpha(n))\int_0^\infty p(n,2,y_2)\eta_2(y_2)dy_2$$

$$+ \beta(n) \int_0^\infty p(n+1,1,y_1)\eta_1(y_1)dy_1.$$

The resulting steady state differential equations for the model are summarized below.

- $s=1;\ n\geq 2,\ y_1>0;$

$$\frac{\partial}{\partial y_1}p(n,1,y_1) = \lambda(n-1)\,p(n-1,1,y_1) \tag{6.1}$$

$$-(\lambda(n)+\eta_1(y_1))\,p(n,1,y_1)$$

- $s=1,\ n=1,\ y_1>0;$

$$\frac{\partial}{\partial y_1}p(1,1,y_1) = -(\lambda(1)+\eta_1(y_1))\,p(1,1,y_1) \tag{6.2}$$

- $s=0,\ n=0;$

$$\lambda(0)p(0,0,0) = \int_0^\infty p(1,1,y_1)\eta_1(y_1)dy_1 \tag{6.3}$$

- $s=2,\ n\geq 2,\ y_2>0;$

$$\frac{\partial}{\partial y_2}p(n,2,y_2) = \lambda(n-1)\,p(n-1,2,y_2) \tag{6.4}$$

$$+(\lambda(n)+\eta_2(y_2))p(n,2,y_2)$$

- $s=1,\ n>1$

$$p(n,1,0) = \alpha(n)\int_0^\infty p(n,2,y_2)\eta_2(y_2)dy_2 + (1-\beta(n))\int_0^\infty p(n+1,1,y_1)\eta_1(y_1)dy_1 \tag{6.5}$$

- $s=1,\ n=1$

$$p(1,1,0) = (1-\beta(1))\int_0^\infty p(2,1,y_1)\eta_1(y_1)dy_1 + \alpha(1)\int_0^\infty p(1,2,ysun2)\eta_2(y_2)dy_2 + \lambda(0)\,p(0,0,0) \tag{6.6}$$

- $s=2,\ n\geq 1.$

$$p(n,2,0) = (1-\alpha(n))\int_0^\infty p(n,2,y_2)\eta_2(y_2)dy_2 + \beta(n)\int_0^\infty p(n+1,1,y_1)\eta_1(y_1)dy_1 \tag{6.7}$$

In the following sections, we apply the equations derived above to solve for networks with a number of stations large enough to be considered infinite and for networks with finite numbers of stations.

For the first type of networks we use the generating function based approach to study the behavior of the stochastic process $N(t)$. To solve for networks with finite numbers of stations, we introduce a linear transformation which reduces the above system of equations into a diagonal form.

6.3.1 Infinite Source Model

In this section, we investigate the stochastic behavior of the process $[N,S,Y]$ when the number of stations is large enough to be considered infinite. We denote by λ the aggregate mean interarrival time of the Poisson process. In order to solve the established set of equations that describes the stochastic behavior of the system, we introduce the generating function $R_s(z,y_s)=\sum_{n=1}^{\infty}p(n,s,y_s)z^n$.

Using equations (6.1) and (6.2) we can write

$$\frac{\partial}{\partial y_1}\sum_{n=1}^{\infty}p(n,1,y_1)z^n=-(\lambda+\eta_1(y_1))\sum_{n=1}^{\infty}p(n,1,y_1)z^n$$

$$+\lambda\sum_{n=1}^{\infty}p(n-1,1,y_1)z^n$$

which could be rewritten as

$$\frac{\partial}{\partial y_1}R_1(z,y_1)=-(\lambda(1-z)+\eta_1(y_1))\,R_1(z,y_1)\,;$$

The solution of the above differential difference equation is

$$R_1(z,y_1)=R_1(z,0)e^{-\lambda y_1(1-z)-\int_0^{y_1}\eta_1(v)dv}\,;$$

Define $R_s(z)=\int_0^{\infty}R_s(z,y_s)dy_s, s=1,2$. Notice that $R_s(z)$, $s=1,2$, represents the generating function of the number of stations currently competing for the channel when the system is in state s. Using the above equation, for $s=1$, we get

$$R_1(z)=R_1(z,0)\int_0^{\infty}e^{-\lambda y_1(1-z)-\int_0^{y_1}\eta_1(v)dv}\,dy_1.$$

Let $g(y_1)=e^{-\int_0^{y_1}\eta_2(v)dv}$ and $\frac{d}{dy_1}h(y_1)=e^{-\lambda y_1(1-z)}$

Using the integration by parts method, we get

$$R_1(z) = R_1(z,0)\,\frac{1}{\lambda(1-z)}[1-\int_0^{\infty} e^{-\lambda(1-z)y_1-\int_0^{y_1}\eta_1(v)dv}\,\eta_1(y_1)dy_1],$$

which can further be written as

$$R_1(z) = R_1(z,0)1\frac{}{\lambda(1-z)}[1-\int_0^{\infty} e^{-\lambda(1-z)y_1}\,b_1(y_1)dy_1];$$

Therefore, the solution of the above differential equation is

$$R_1(z) = \frac{(1-B_1^*(\lambda(1-z)))}{\lambda(1-z)}R_1(z,0)\,;$$

Similarly, using the fact that

$$\frac{\partial}{\partial y_2}R_2(z,y_2) = -(\lambda(1-z)+\eta_2(y_2))R_2(z,y_2)$$

we get

$$R_2(z) = \frac{(1-B_2^*(\lambda(1-z)))}{\lambda(1-z)}R_2(z,0).$$

where $B_1^*(x)$ and $B_2^*(x)$ are the Laplace transform of the transmission time, $b_1(t)$, and the contention step time, $b_2(t)$, respectively.

Notice that R_s(z,0), s=1,2, are still unknown and will be determined using the boundary conditions. In the next section we present the closed form solution of these equations when the probability of collision and the probability of resolution are assumed to be load independent. The conditions for existence and uniqueness of the solution are also provided.

6.3.1.1 Constant functions

In following section, we solve the model assuming that the probability of conflict and the probability of resolution are independent of the number of stations in the system; i.e., $\alpha(n)=\alpha$ and $\beta(n)=\beta$. Let $R_1(z,0)=\sum_{n=1}^{\infty}p(n,1,0)z^n$; Using (6.3), (6.4) and (6.5), we get

$$\sum_{n=1}^{\infty}p(n,1,0)z^n=(1-\beta)\int_0^{\infty}\sum_{n=1}^{\infty}z^n p(n+1,1,y_1)\eta_1(y_1)dy_1$$

$$+\alpha\int_0^{\infty}\sum_{n=1}^{\infty}p(n,2,y_2)\eta_2(y_2)dy_2 z^n+\lambda z\,p(0,0,0),$$

which yields

$$R_1(z,0)=\int_0^{\infty}\frac{(1-\beta)}{z}[R_1(z,y_1)\eta_1(y_1)dy_1]-(1-\beta)\int_0^{\infty}p(1,1,y_1)\eta_1(y_1)dy_1$$

$$+\alpha\int_0^{\infty}R_2(z,y_2)\eta_2(y_2)dy_2+\lambda z\,p(0,0,0)\,;$$

Therefore

$$R_1(z,0)=\frac{(1-\beta)}{z}[R_1(z,0)B_1^*(\lambda(1-z))]-\lambda(1-\beta)\,p(0,0,0)$$

$$+\alpha R_2(z,0)B_2^*(\lambda(1-z)).+\lambda z\,p(0,0,0).$$

Similarly, we can solve for $S(t)=2$; we get

$$\sum_{n=1}^{\infty}p(n,2,0)z^n=(\int_0^{\infty}\sum_{n=1}^{\infty}p(n+1,1,y_1)z^n\eta_1(y_1)dy_1)\,\beta$$

$$+(\int_0^{\infty}\sum_{n=1}^{\infty}z^n p(n,2,y_2)\eta_2(y_2)dy_2)(1-\alpha),$$

which yields

$$R_2(z,0) = \frac{\beta}{z}[R_1(z,0)B_1^*(\lambda(1-z))] - \lambda\beta\, p(0,0,0)$$

$$+ (1-\alpha)R_2(z,0)B_2^*(\lambda(1-z))$$

The resulting system of equations is

$$R_1(z,0) = \frac{\alpha z B_2^*(\lambda(1-z))}{(z-(1-\beta)B_1^*(\lambda(1-z))}R_2(z,0) \;+\; \frac{\lambda z(z-(1-\beta))}{(z-(1-\beta)B_1^*(\lambda(1-z))}p(0,0,0),$$

$$R_2(z,0) = \frac{\beta B_1^*(\lambda(1-z))}{z(1-(1-\alpha)B_2^*(\lambda(1-z))}R_1(z,0) \;-\; \frac{\lambda\beta}{(1-(1-\alpha)B_2^*(\lambda(1-z))}p(0,0,0)\ .$$

Solving the above system of linear equations we get

$$R_1(z,0) = \frac{\gamma_1(z)}{\gamma(z)}p(0,0,0)\,;$$

$$R_2(z,0) = \frac{\gamma_2(z)}{\gamma(z)}p(0,0,0).$$

where

$$\gamma(z) = z-(1-\beta)B_1^*(\lambda(1-z))-z(1-\alpha)B_2^*(\lambda(1-z))$$

$$+ (1-\alpha-\beta)B_2^*(\lambda(1-z))B_1^*(\lambda(1-z)))\,;$$

$\gamma_1(z) = \lambda z((z-(1-\beta))+((1-\alpha)(1-z)-\beta)B_2^*(\lambda(1-z)))$;and

$$\gamma_2(z) = -\lambda\beta z(1-B_1^*(\lambda(1-z)))\,.$$

Let

$$Q(z) = p(0,0,0) + R_1(z) + R_2(z)$$

be the generating function of the number of stations currently requesting access to the channel. Using the above derivations, $Q(z)$ can be written as

$$Q(z) = \frac{(1-B_1^*(1-\lambda(1-z)))}{\lambda(1-z)} R_1(z,0)$$

$$+ \frac{(1-B_2^*(1-\lambda(1-z)))}{\lambda(1-z)} R_2(z,0) + p(0,0,0);$$

Notice that $p(0,0,0)$ is determined by the necessary condition that $Q(1)=1$. Using l'Hˆospital's rule we obtain

$$p(0,0,0) = \frac{\alpha-\lambda(\beta\overline{b_2}+\alpha\overline{b_1})}{\alpha-\lambda\beta\overline{b_2}} = 1-\frac{\alpha\rho_1}{\alpha-\beta\rho_2}$$

where $\rho_s = \lambda\overline{b_s}$, $s=1,2$;

The necessary condition that $0<p(0,0,0)<1$ yields

$$\alpha-\beta\rho_2>0 \qquad \text{(i)}$$

$$\rho_1+\frac{\beta}{\alpha}\rho_2 < 1 \qquad \text{(ii)}$$

The following theorem proves our claim that the above conditions are sufficient for the system to be ergodic.

Theorem

Let N_i be the number of messages in the network after the *ith* transmission. We denote by T_i the instant at which the *ith*-transmission ends. Consider the Markov chain embedded in the process Z_t at $t=T_i$. The aperiodic irreducible Markov chain N_i is ergodic if and only if the conditions (*i*) and (*ii*) are satisfied.

Proof

Conditions (*i*) and (*ii*) are necessary since p(0,0,0) must be a strictly positive quantity less than 1. To show the sufficiency of these conditions, we need to prove that [Pak69]

$| E[N_{i+1}-N_i \mid N_i=n]| < \infty$, for *all integers n,* (a)

$\lim sup_{i\to\infty} E[N_{i+1}-N_i \mid N_i=n] < 0;$ (b)

Let $A_{]t_1,t_2]}$ and $D_{]t_1,t_2]}$ represent the number of messages generated and the number of messages transmitted between t_1 and t_2, respectively. We can write

$N_{i+1}-N_i = A_{]T_i,T_{i+1}]} - D_{]T_i,T_{i+1}]}$, *or*

$E[N_{i+1}-N_i] = E[A_{]T_i,T_{i+1}]}] - E[D_{]T_i,T_{i+1}]}]$

Note that,

$$E[A_{]T_i,T_{i+1}]} \mid N_i = n] = \lambda E[T_{i+1}-T_i \mid N_i = n]\,,$$

and

$$D_{]T_i,T_{i+1}]} = E[D_{]T_i,T_{i+1}]}] = 1.$$

Evaluating $E[T_{i+1}-T_i \mid N_i = n]$, we obtain

$$E[T_{i+1}-T_i \mid N_i = 0] = \frac{1}{\lambda}+\overline{b_1}$$

$$E[T_{i+1}-T_i \mid N_i > n] = (1-\beta)\overline{b_1} + \beta E[T_{i+1}-T_i \mid S(T_i^\dagger) = 2]$$

$$E[T_{i+1}-T_i \mid S(T_i^\dagger) = 2] = \sum_{k=1}^{\infty}(1-\alpha)^{n-1}\alpha(n\overline{b_2} + \overline{b_1}) = \frac{\overline{b_2}}{\alpha} + \overline{b_1}$$

Therefore, $\lim sup_{i\to\infty} E[N_{i+1}-N_i \mid N_i = n] = \lambda(\overline{b_1}+\frac{\beta}{\alpha}\overline{b_2})-1 = \rho_1+\frac{\beta}{\alpha}\rho_2-1.$ This proves the sufficiency of conditions (*i*) and (*ii*) and completes the proof □.

Differentiating $Q(z)$ with respect to z and evaluating the resulting expression at $z=1$ yields the value of the mean number of contending stations in the system, $\overline{N}$. We obtain

$$\overline{N} = \frac{(((2-2\alpha)\overline{b_1 b_2}^2-(\alpha\overline{b_1^2}-2\alpha\overline{b_1}^2)\overline{b_2}+\alpha\overline{b_1}\,\overline{b_2^2})\beta\lambda^3-(2\alpha\overline{b_1 b_2}\beta+2\alpha^2\overline{b_1}^2-\alpha^2\overline{b_1^2})\lambda^2+2\alpha^2\overline{b_1}\lambda)}{((2\overline{b_2}^2\beta^2+2\alpha\overline{b_1 b_2}\beta)\lambda^2+(-4\alpha\overline{b_2}\beta-2\alpha^2\overline{b_1})\lambda+2\alpha^2)}$$

Notice that the entrance in state 0 constitutes a set of regeneration points. Therefore, we can apply Little's law to compute the average time, T, a contending station spends in the system, waiting and transmitting. We obtain

$$T=\frac{(((2-2\alpha)\overline{b_1}\,\overline{b_2}^{2}-(\alpha\overline{b_1^2}-2\alpha\overline{b_1}^{2})\overline{b_2}+\alpha\overline{b_1}\,\overline{b_2^2})\beta\lambda^2-(2\alpha\overline{b_1}\,\overline{b_2}\beta+2\alpha^2\overline{b_1}^{2}-\alpha^2\overline{b_1^2})\lambda+2\alpha^2\overline{b_1})}{((2\overline{b_2}^{2}\beta^2+2\alpha\overline{b_1}\,\overline{b_2}\beta)\lambda^2+(-4\alpha\overline{b_2}\beta-2\alpha^2\overline{b_1})\lambda+2\alpha^2)}$$

6.3.2 Finite Source Model

In the following section, we present the Markovian analysis of the finite source model. First, we notice that if there are n stations currently competing for the channel access and M is the total number of stations in the network, the probability of a newly generated message in the interval $(t,\ t{+}dt)$ is $\lambda(M-n)dt+o(dt)$. Therefore, using matrix notation, the set of equations describing the finite source model can be written as

$$(\Delta_s\mathrm{I}-\mathrm{Q})\mathrm{P}(s,y_s)=0;\quad s=1,2; \qquad (6.8)$$

where

$$\mathrm{P}(s,y_s)=\begin{bmatrix} p(1,s,y_s)\\ p(2,s,y_s)\\ \\ \cdot\\ \cdot\\ \\ p(M,s,y_s)\end{bmatrix};$$

$$\mathrm{Q}=\begin{bmatrix}
-\lambda(M-1) & 0 & 0 & \cdot & & \cdot & \cdot & & 0\\
\lambda(M-1) & -\lambda(M-2) & 0 & \cdot & & \cdot & \cdot & & 0\\
0 & \lambda(M-2) & -\lambda(M-3) & \cdot & & \cdot & \cdot & & 0\\
\cdot & \cdot & \cdot & & & & & & \cdot\\
\cdot & \cdot & \cdot & & & & & & \cdot\\
\cdot & \cdot & \cdot & \lambda(M-(i-1)) & -\lambda(M-i) & 0 & & & \cdot\\
 & & & & & & & & \\
 & & & & & & 2\lambda & -\lambda & 0\\
0 & 0 & 0 & \cdot & & \cdot & 0 & \lambda & 0
\end{bmatrix}$$

and Δ_s is the operator $\dfrac{\partial}{\partial y_s}+\eta(y_s)$ and I is the $M\times M$ identity matrix.

Similarly, the boundary conditions can be written as

$$P(1,0)=\int_0^\infty C_1P(1,y_1)\eta_1(y_1)dy_1 \tag{6.9}$$

$$+\int_0^\infty C_2P(2,y_2)\eta_2(y_2)dy_2+\lambda MC_0p(0,0,0).$$

$$P(2,0)=\int_0^\infty \overline{C_1}P(1,y_1)\eta_1(y_1)dy_1 \tag{6.10}$$

$$+\int_0^\infty \overline{C_2}P(2,y_2)\eta_2(y_2)dy_2.$$

where

$$C_0=\begin{bmatrix}1\\0\\0\\ \cdot\\ \cdot\\ \\ 0\end{bmatrix};\quad C_1=\begin{bmatrix}0 & (1-\beta(1)) & 0 & \cdot & 0\\ 0 & 0 & (1-\beta(2)) & \cdot & 0\\ & & 0 & & 0\\ \cdot & \cdot & & 0 & \cdot\\ \cdot & \cdot & \cdot & (1-\beta(i)) & \cdot\\ & & & 0 & \cdot\\ & & & \cdot & (1-\beta(M-1))\\ 0 & \cdot & \cdot & \cdot & 0\end{bmatrix}$$

$$C_2=\begin{bmatrix}\alpha(1) & 0 & 0 & \cdot & \cdot & 0\\ 0 & \alpha(2) & 0 & \cdot & \cdot & 0\\ \cdot & 0 & \alpha(3) & \cdot & \cdot & 0\\ \cdot & & 0 & 0 & & \cdot\\ & & \cdot & \alpha(i) & & \cdot\\ & & & & \alpha(M-1) & 0\\ 0 & \cdot & & & 0 & \alpha(M)\end{bmatrix}$$

and

$$\overline{c_{ij}}=\begin{cases}1-c_{ij} & \text{if } c_{ij}>0\\ 0 & \textit{otherwise}\end{cases}$$

Due to the presence of variable coefficients, the generating function approach used to solve the infinite source model cannot be easily applied to the finite source model. To solve these equations we need to determine a linear transformation which reduces the above system of equations into a diagonal form.

Let T and its inverse T^{-1} be the left and right matrices of eigenvectors of Q. We know that

$$Q = T^{-1}\Theta T \tag{6.11}$$

where Θ is a diagonal matrix whose elements, $\theta_i (i=1,...N)$, are the eigenvalues of Q; that is

$$\Theta = \begin{bmatrix} \theta_1 & 0 & . & . & & & 0 \\ 0 & \theta_2 & 0 & & . & . & 0 \\ . & 0 & \theta_3 & & . & . & 0 \\ . & & 0 & 0 & & & . \\ & & . & \theta_i & & & . \\ & & & & & \theta_{M-1} & 0 \\ 0 & . & & & & 0 & \theta_M \end{bmatrix}$$

and θ_i is the *ith* eigenvalue of Q. Substituting (6.11) in (6.8), we get $(\Delta_s I - \Theta)L(s,y_s)=0$ where

$$L(s,y_s)=TP(s,y_s), \quad s=1,2.$$

The solution of the above equation is given by $L(s,y_s)=\Lambda_s L(s,0)e^{-\int_0^{y_s}\eta_s(v)dv}$, where

$$\Lambda_s = \begin{bmatrix} e^{y_s\theta_1} & 0 & 0 & . & . & . & 0 \\ 0 & e^{y_s\theta_2} & 0 & . & . & . & 0 \\ . & 0 & e^{y_s\theta_3} & . & . & . & 0 \\ . & & 0 & 0 & & & . \\ & & . & e^{y_s\theta_i} & & & . \\ & & & 0 & e^{y_s\theta_{M-1}} & & . \\ 0 & . & & & 0 & & e^{y_s\theta_M} \end{bmatrix}$$

L(s, 0) is still unknown and will be determined using the boundary conditions. Applying the above linear transformation to (6.9) results in

$$P(1,0)=\int_0^\infty C_1T^{-1}L(1,y_1)\eta_1(y_1)dy_1$$

$$+\int_0^\infty C_2T^{-1}L(2,y_2)\eta_2(y_2)dy_2+\lambda M C_0p(0,0,0).$$

Multiplying both sides of the above equation by **T**, we obtain

$$L(1,0)=TC_1T^{-1}\int_0^\infty L(1,y_1)\eta_1(y_1)dy_1$$

$$+TC_2T^{-1}\int_0^\infty L(2,y_2)\eta_2(y_2)dy_2+\lambda M C_0p(0,0,0).$$

Replacing L(s,y_s) ;s=1,2 in the above equation, we get

$$L(1,0)=TC_1T^{-1}\int_0^\infty \Lambda_1L(1,0)e^{-\int_0^{y_1}\eta_1(u)du}\eta_1(y_1)dy_1$$

$$+TC_2T^{-1}\int_0^\infty \Lambda_2L(2,0)e^{-\int_0^{y_2}\eta_1(u)du}\eta_2(y_2)dy_2+\lambda MTC_0p(0,0,0).$$

Let $B_s^*(x)$ be the Laplace transform of $b_s(x)$;s=1,2 , the above equation reduces to

$$L(1,0)=TC_1T^{-1}\tilde{B}_1^*L(1,0)+TC_2T^{-1}\tilde{B}_2^*L(2,0)+\lambda MTC_0p(0,0,0).$$

where the matrix $\tilde{B}_s^*$ is of the form

$$\tilde{B}_s^*=\begin{bmatrix} B_s^*(\theta_1) & 0 & 0 & . & . & 0 \\ 0 & B_s^*(\theta_2) & 0 & . & . & 0 \\ . & 0 & B_s^*(\theta_3) & . & . & . \\ . & & 0 & & & \\ & & . & B_s^*(\theta_i) & & \\ & & & & B_s^*(\theta_{M-1}) & \\ 0 & . & & & 0 & B_s^*\theta_M) \end{bmatrix}$$

Similarly, (6.10) becomes

$$L(2,0)=T\overline{C_1}T^{-1}\tilde{B}_1^*L(1,0)+T\overline{C_2}T^{-1}\tilde{B}_2^*L(2,0)$$

The above set of equations determines $L(s, 0)$ and hence $L(s,y_s)$; $s=1,2$.

In the remaining of this section, we will show the corresponding results for the case where the probability of collision and the probability of resolution are constant.

6.3.2.1 Constant functions

In the section, we present the solution of the above equations in the case where $\alpha(n)$ and $\beta(n)$ are independent of the current number of stations in the system. We first define the discrete transform which converts the set of the above equations into a mathematically more tractable set of equations. We then use the transformed equations to compute the system state probabilities.

The appropriate discrete transform, which reduces the system of equations to a linear form is defined by

$$l(n,s,y_s) = \sum_{j=n}^{M-1} \begin{bmatrix} j \\ n \end{bmatrix} p(M-j,s,y_s), \qquad (0 \le n < M),(s=1,2)$$

or

$$p(n,s,y_s) = \sum_{i=0}^{n-1} (-1)^i \begin{bmatrix} M-n+i \\ i \end{bmatrix} l(M-n+i,s,y_s),(1 \le n \le M),(s=1,2) \qquad (6.12)$$

where $\begin{bmatrix} j \\ n \end{bmatrix}$ is the binomial coefficient.

Using the above transform , the steady state differential equations reduce to

$$\frac{\partial}{\partial y_s} l(n,s,y_s) = -(\lambda n + \eta_s(y_s)) l(n,s,y_s), \qquad (y_s>0),(s=1,2)$$

The solution of this differential equation is given by $l(n,s,y_s)=l(n,s,0)e^{-n\lambda y_s - \int_0^{y_s} \eta_s(u)du}$;

Similarly, applying the same transform to the boundary conditions and substituting $l(n,s,y_s)$ by their values results into the following system of equations

$$l(n,1,0) = (1-\beta)l(n,1,0)B_1^*(\lambda n) \tag{6.13}$$

$$+ (1-\beta)l(n-1,1,0)B_1^*((n-1)\lambda) + \alpha l(n,2,0)B_2^*(n\lambda)$$

$$+\lambda M\left[\begin{bmatrix} M-1 \\ n \end{bmatrix} - (1-\beta)\begin{bmatrix} M \\ n \end{bmatrix}\right]p(0,0,0)$$

$$l(n,2,0) = \beta l(n,1,0)B_1^*(n\lambda) \tag{6.14}$$

$$+ \beta l(n-1,1,0)B_1^*((n-1)\lambda) + (1-\alpha)l(n,2,0)B_2^*(n\lambda)$$

$$- \lambda M\begin{bmatrix} M \\ n \end{bmatrix}\beta p(0,0,0)$$

Solving the above linear system of equations, we obtain

$$l(n,1,0) = a(n)l(n-1,1,0) + b(n)p(0,0,0), \qquad (0<n<M-1)$$

where

$$a(n) = \frac{((1-\beta)+\alpha\phi(n)B_2^*(\lambda n))B_1^*(\lambda(n-1))}{(1-B_1^*(\lambda n)((1-\beta)+\alpha\phi(n)B_2^*(\lambda n)))}$$

$$b(n) = \frac{\lambda M\left[\begin{bmatrix} M-1 \\ n \end{bmatrix} - \begin{bmatrix} M \\ n \end{bmatrix}((1-\beta)+\alpha\phi(n)B_2^*(\lambda n))\right]}{(1-B_1^*(\lambda n)((1-\beta)+\alpha\phi(n)B_2^*(\lambda n)))}$$

and

$$\phi(n) = \frac{\beta}{(1-(1-\alpha)B_2^*(n\lambda))}$$

Substituting $l(n,1,0)$ in (6.14) we obtain

$$l(n,2,0) = \frac{\beta}{((1-\beta)+(\alpha+\beta-1)B_2^*(\lambda n))}l(n,1,0)$$

$$-\frac{\lambda M\beta\begin{bmatrix} M-1 \\ n \end{bmatrix}}{((1-\beta)+(\alpha+\beta-1)B_2^*(\lambda n))}p(0,0,0), \qquad (0\le n<M)$$

The above recursive equations are to be solved subject to the boundary condition

$$p(1,2,0) = 0 \tag{6.15}$$

This condition states the fact that a conflict does not occur when only one station is in the system. Using equation (6.12), the above condition reduces to

$$l(M-1,2,0)=0,$$

or

$$l(M-1,1,0)=\lambda Mp(0,0,0) \qquad (6.16)$$

The above equation completely determines $l(n,s,0)$. Notice that $p(0,0,0)$ is determined by the necessary condition

$$\sum_{s=1}^{2}\sum_{n=1}^{M} p(n,s) + p(0,0,0)=1 \qquad (6.17)$$

Let $p(n,s)=\int_0^{\infty} p(n,s,y_s)dy_s, s=1,2$. Using (6.12) we obtain

$$p(n,s)=\sum_{i=0}^{n-1}(-1)^i\begin{bmatrix}M-n+i\\ i\end{bmatrix} l(M-n+i,s,0)r_s(M-n+i)$$

where

$$r_s(k)=\begin{cases}\dfrac{B_s^*(k)}{k\lambda} & \text{if } k>0\\ \overline{b_s} & \text{if } k=0\end{cases}$$

Let $\Pi_s(z)=\sum_{n=1}^{M} z^n p(n,s)$ be the generating function of the number of stations currently attempting to access the channel when the system is in state s.

$$\Pi_s(z)=\sum_{n=1}^{M} z^n \sum_{i=0}^{n-1}(-1)^i\begin{bmatrix}M-n+i\\ i\end{bmatrix} l(M-n+i,s,0)r_s(M-n+i)$$

Define $Q(z)=\Pi_1(z)+\Pi_2(z)+p(0,0,0)$ to be the generating function of the number of stations currently attempting to access the channel. The mean number of contending stations can be obtained from the above equation by differentiating $Q(z)$ with respect to z and setting $z=1$. In the next section we will develop a set of experiments to study the effect of α and β on the performance of a channel multiaccess protocol.

6.4 EXPERIMENTAL RESULTS

A series of numerical experiments were conducted to investigate the impact of the two parameters α and β, the service distribution and the size of the network on the performance of a multiaccess protocol.

The input parameters include the channel bandwidth, the size of the messages, the parameters controlling the distribution of the network load among different stations and the length of the resolution step. In all experiments, the channel transmission rate is assumed to be 10 Mbps and the resolution step to be equal to the channel round-trip delay. The frame size was either fixed to be 1000 bits or drawn from an exponential distribution with mean 1000.

In Figure 6.2, the value of the network delay of the infinite source model is plotted against the network traffic, $\rho=\lambda\overline{b_1}$, for different values of α. It should be noted that the probability of conflict, β, was chosen to be an increasing function of the network load. In this figure, we assume that the transmission time is exponentially distributed and the step time is constant. The results indicate that if ρ is less than 0.6, the increase in the access delay is not significant. However, the increase in the delay becomes noticeable when ρ approaches one and α is less than 0.3.

In Figure 6.3-(a) and 6.3-(b), the mean number of contending stations is plotted against the network load for a finite number of stations and for different values of α. In the first experiment, Figure 6.3-(a), the network load varied from 0.1 to 0.9, while in the second experiment, Figure 6.3-(b), the load varied from 1.0 to 2.0. The results indicate that the mean number of contending stations increases as ρ increases and the probability of resolution α decreases. However, the increase of the mean number of contending stations becomes significant as ρ approaches one and α becomes small. The same effect

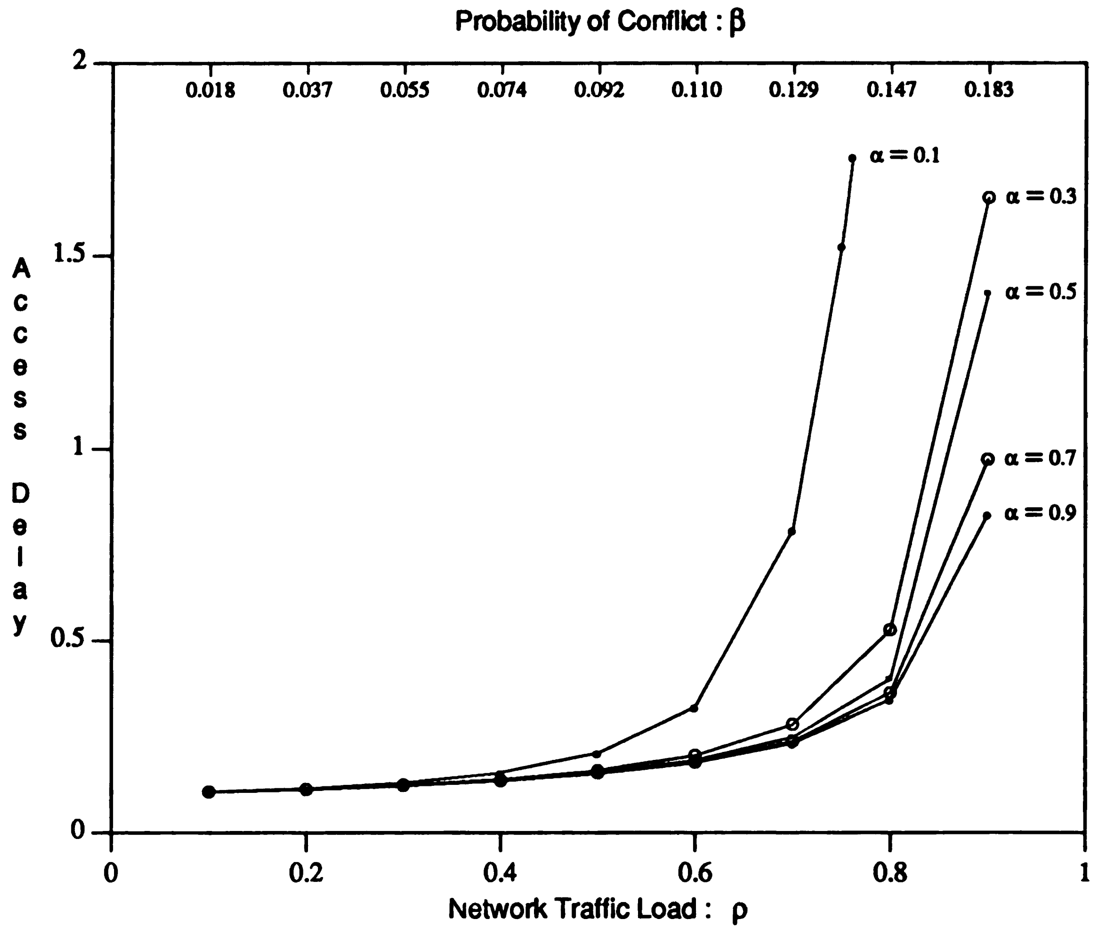

Figure 6.2 Effect of the Load, Resolution and Conflict Probabilities on the Access Delay

(Infinite Source Model)

of the probability of conflict and probability of resolution on the mean queue length of contending stations is observed when ρ ranged between 1.0 and 2.0.

In Figure 6.4, the value of the network access, for both deterministic and exponential transmission time distribution, is plotted against the network traffic for an infinite number of stations. The probability of resolution was fixed to 0.8 and the probability of conflict was an increasing function of the network load. The figure shows also the results

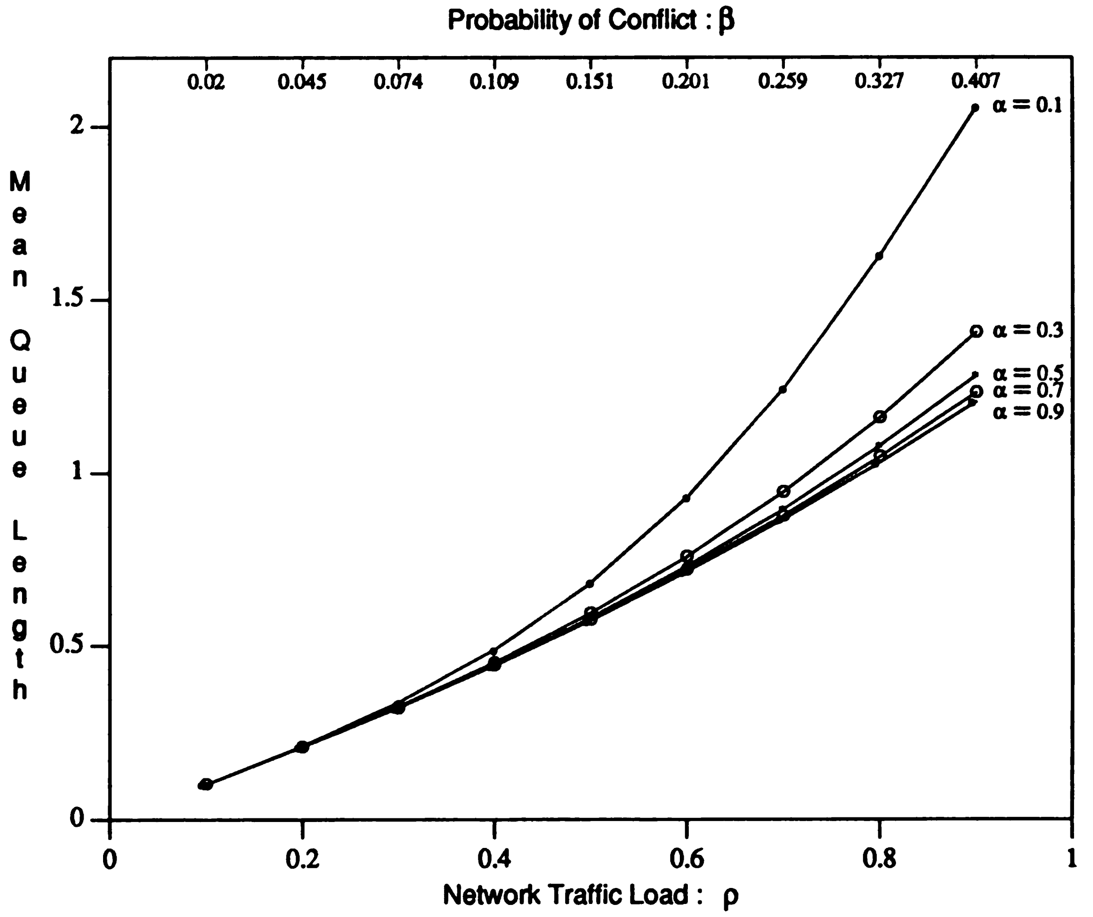

Figure 6.3-(a) Effect of the Load, Resolution and Conflict Probabilities on the Mean Queue Length

(Finite Source Model)

for the corresponding standard M/D/1 and M/M/1 queues. The results indicate that the effect of the variability of the transmission time is predominant when the network load, $\rho = \lambda \overline{b_1}$, is near one. The results also show that the overhead incurred by the protocol is not significant for small values of ρ. However, a significant increase in the access delay is observed when ρ approaches 1.0.

Figures 6.5-(a) and Figures 6.5-(b) show the effect of the transmission time variability on the mean number of contending stations for different network loads. The results indicate that the transmission time variability is not significant for ρ<<1.0 or when

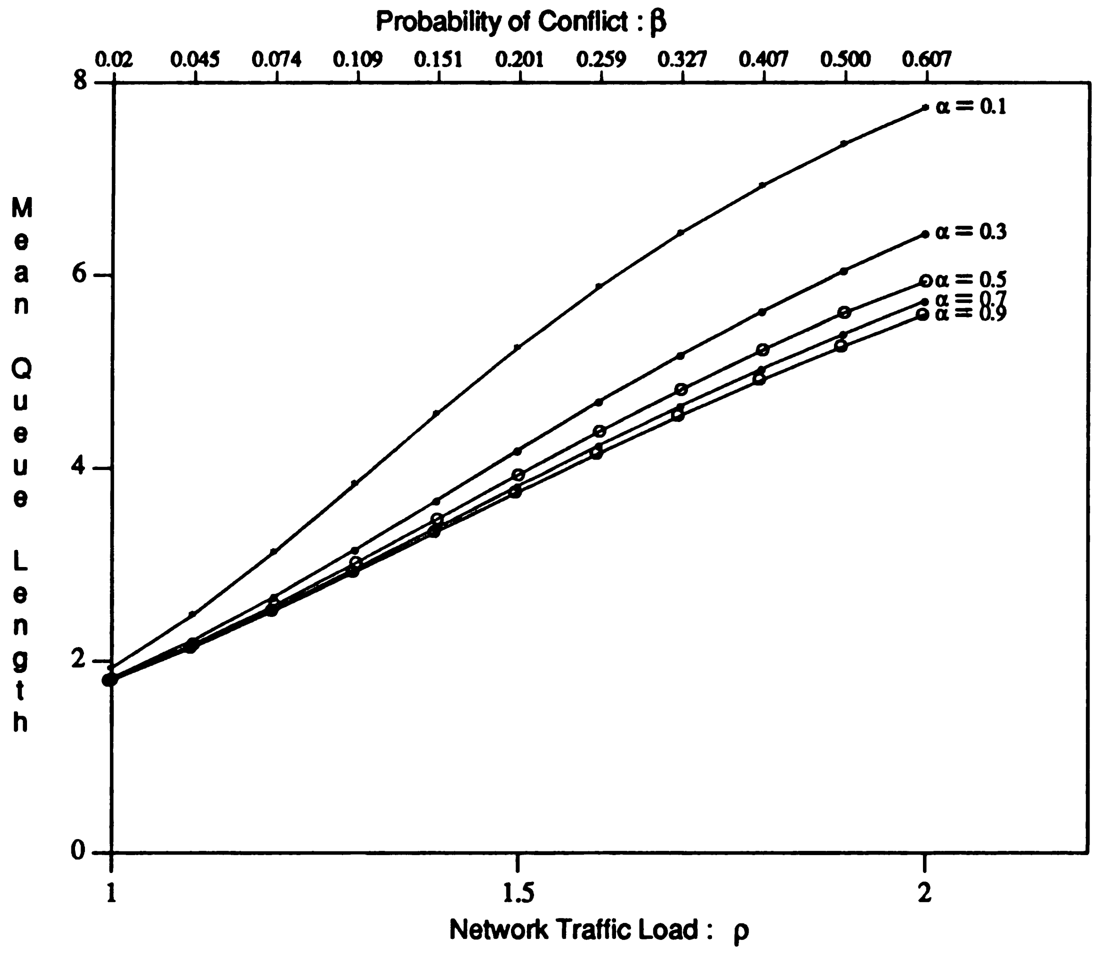

Figure 6.3-(b) Effect of the Load, Resolution and Conflict Probabilities on the Mean Queue Length

(Finite Source Model)

$\rho >> 1.0$. However, when $\rho \approx 1.0$, the effect becomes significant.

The last experiment aimed at studying the effect of the size of the network on the mean number of contending stations for different values of ρ. The results are recorded in Figure 6.6. The results indicate that for a given network load the mean number of contending stations increase as the size of the network increases. The increase becomes significant as ρ approaches one. Based on the above observations, the use of the simple infinite source model is bound to be erroneous if the size of the network is not

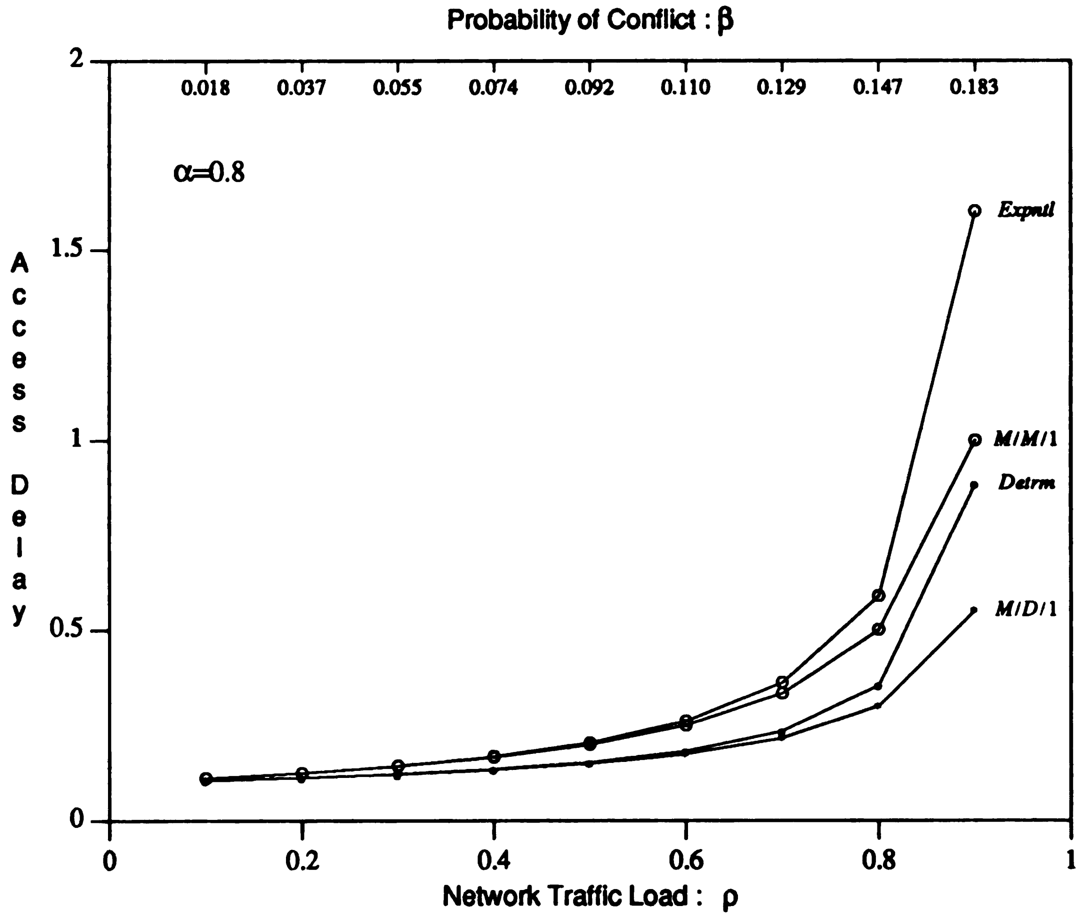

Figure 6.4 Effect of the Load and the Message Length Distribution on the Access Delay (Infinite source model)

sufficiently large.

From the above discussion we conclude that for a network with the specifications described above, the probability of resolution becomes predominant when its values decreases below 0.3 and ρ approaches one. We can also that the effect of the transmission time variability becomes predominant as ρ approaches one. In this region, the use of the simple infinite source model to study the performance of a protocol when the number of stations is small does not produce accurate results.

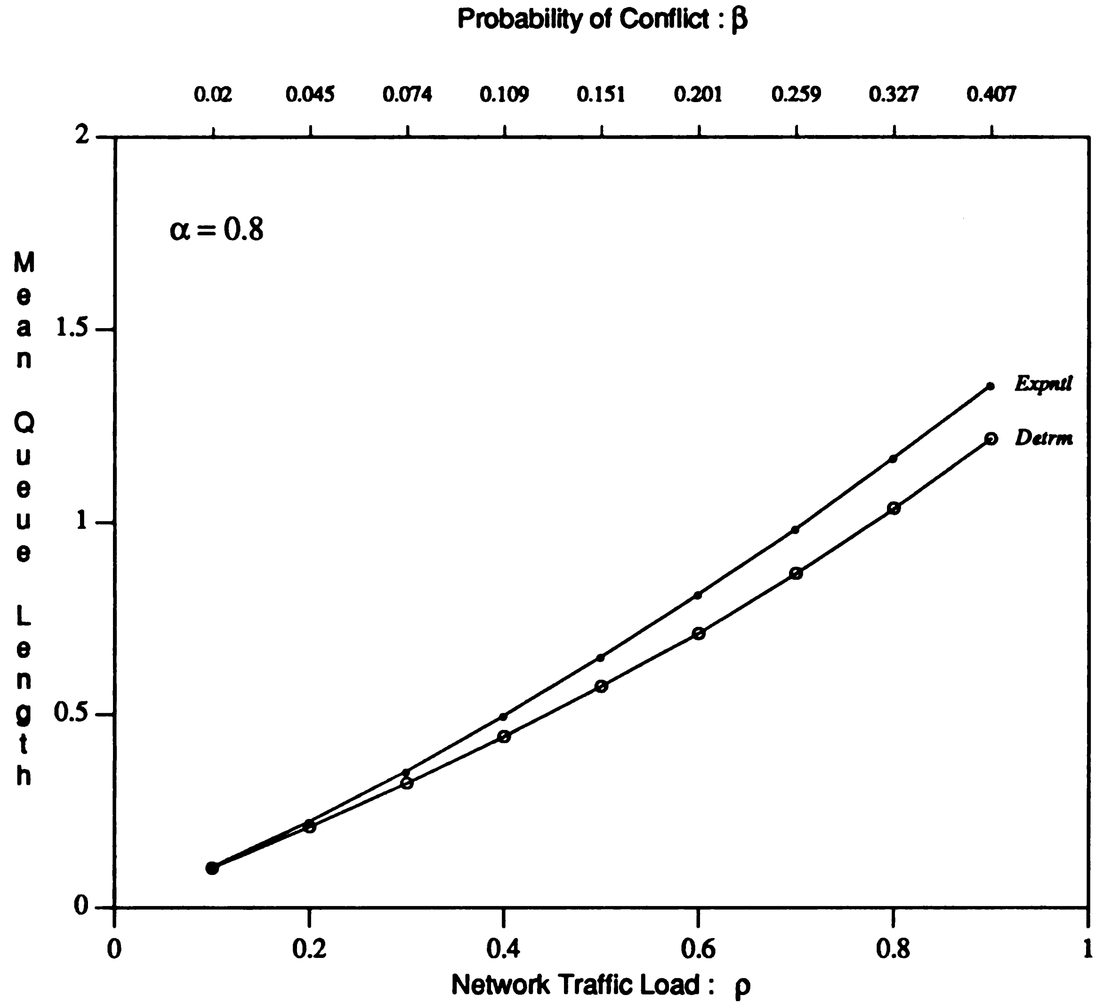

Figure 6.5-(a) Effect of the Load and the Message Length Distribution on the Mean Queue Length

(Finite Source Model)

6.5 MODEL APPLICATION

Our objective in the remainder of this chapter is to apply the analytical model to solve for the average message response time of the virtual token protocol described in earlier chapters.

In the previous sections, we developed the expression for the average message delay as a function of the probability of conflict β, and the probability of resolution α. Our

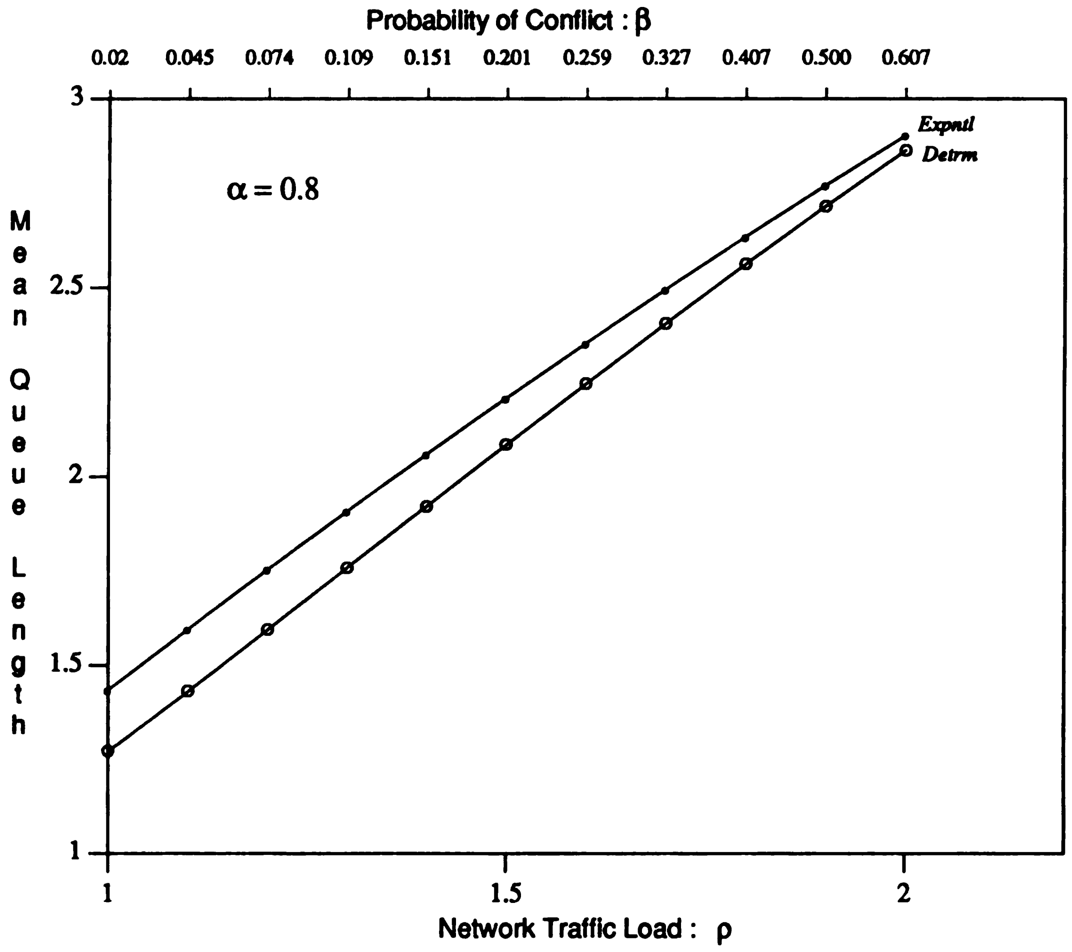

Figure 6.5-(b) Effect of the Load and the Message Length Distribution on the Mean Queue Length

(Finite Source Model)

analysis problem therefore simply reduces to the calculations of these probabilities. The derivation of these probabilities is the object of the following sections.

In the first part of the discussion, we introduce the appropriate notations and state the assumptions that define our model of computation. In the second part, we proceed with the computation of β and α.

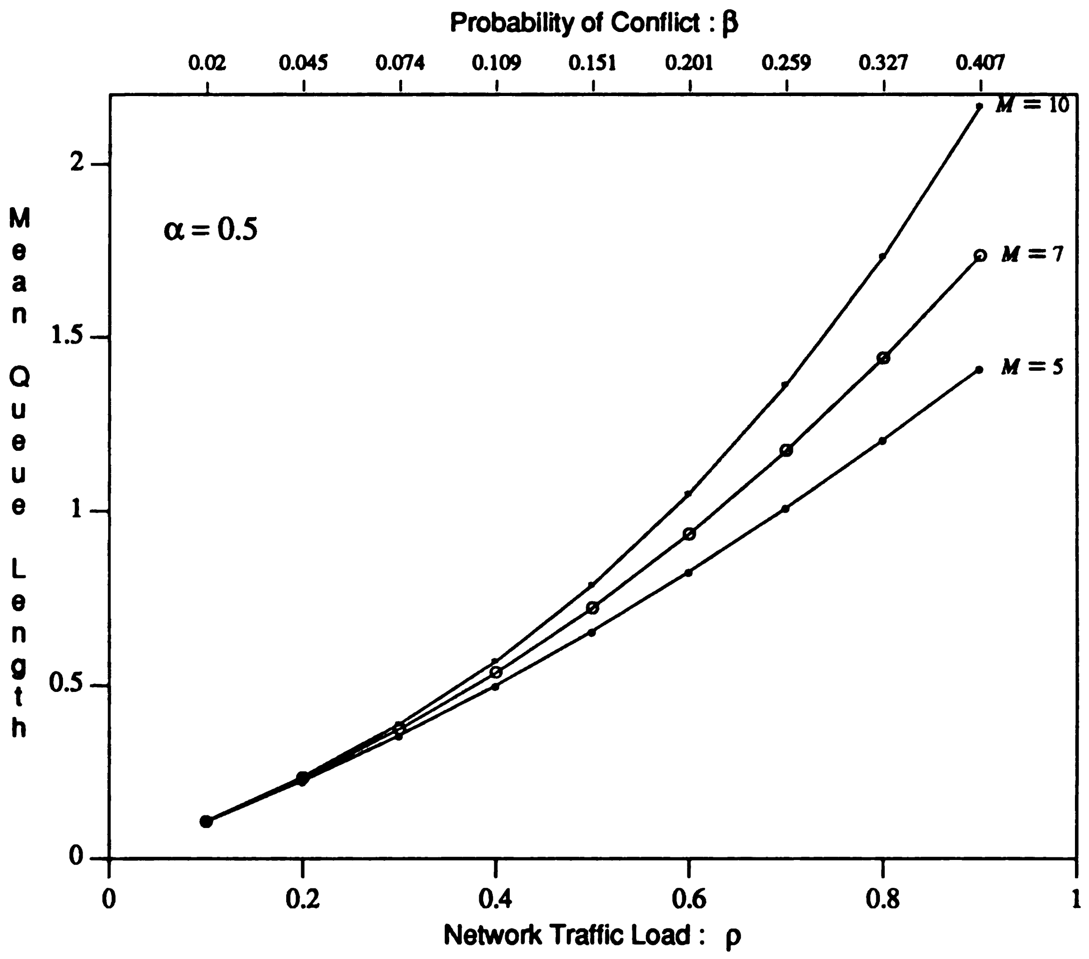

Figure 6.6 Effect of the Load and the Network Size on the Mean Queue Length

(Finite Source Model)

6.5.1 Notation and Model Assumptions

In this model, the channel is assumed to be noiseless and perfectly reliable. Traffic entering the network forms a *Poisson* process. We use λ_{ij} to denote the arrival rate of class j messages, $j=1,\ldots m-1$, to station i ,$i=0,1\ldots M-1$. We assume that all classes of messages have lengths that are either constant or drawn independently from an exponential distribution. We also assume that the PAP search method used is based on the static dichotomizing approach. We further define:

- The total traffic entering the network as $\lambda = \sum_{i=0}^{M-1} \sum_{j=1}^{m-1} \lambda_{ij}$,
- The total traffic of priority j, $1 \le j \le m-1$, entering the network as $\lambda_j^p = \sum_{i=0}^{M-1} \lambda_{ij}$, and
- The traffic intensity entering a given station i, $0 \le i \le M-1$, as $\lambda_i^s = \sum_{j=0}^{m-1} \lambda_{ij}$.

6.5.2 Probability Evaluation

In this section, we compute the values of β and α. We begin by deriving the probability, $\gamma = 1-\beta$, that no conflict occurs after the transmission of a message. We then compute the probability of resolution α.

As stated in the previous chapter, the probability, $c_i(p)$, that station i, $0 \le i < M-1$, generates a contention parameter with priority p is given by

$$\begin{cases} c_i(p) = \rho_{ip} \prod_{j=p+1}^{m-1} (1-\rho_{ij}) & 1 \le p < m-1 \\ \rho_{im-1} & p = m-1 \end{cases}$$

We use $C_i(p) = \sum_{k=1}^{p} c_i(p)$ to denote the distribution of the above density function.

Let $\gamma_i(j)$ be the probability that no collision occurs, given that the next virtual token holder is station i and the network priority is j. Station i successfully transmits if no station other than itself has a priority higher than j. Therefore, $\gamma_i(j)$ can be written as:

$$\gamma_i(j) = (1-C_i(j)) \prod_{\substack{k=0 \\ k \ne i}}^{M-1} \left(\prod_{p=1}^{m-1} (1-\rho_{kp}) + C_k(j)\right)$$

The probability, $\gamma(j)$, that no collision occurs at the end of the current message transmission given that the network priority is j can be computed as:

$$\gamma(j) = \sum_{i=0}^{M-1} \gamma_i(j) \frac{\lambda_{ij}}{\lambda_j^p}$$

Consequently, the probability, γ , that no collision occurs at the end of the current message service is given by:

$$\gamma = \sum_{j=1}^{m-1} \gamma(j) \frac{\lambda_j^p}{\lambda}$$

The above equations completely determine the probability of conflict $\beta = 1 - \gamma$.

We now consider the derivation of α, the probability of resolution. We define the following random variables.

H : the identity of the station currently holding the virtual token,

v_i : The value of the contention parameter of a given station i, $0 \le i < M$.

Prior to obtaining the value of α, we derive the distribution function of the contention parameter value. More specifically, we need to compute the probability $V_i^h(v) = Pr[\, v_i > v \mid H = h \,]$, where $v = (p-1)M + r$, $0 \le r < M$ and $0 < p < m$, is a given value of a contention parameter.

It is clear that the contention parameter of station i is higher than $(p-1)M + r$ if and only if :

- Station i has a higher priority than p, or
- Station i has a priority equal to p and i is closer to station h than station $(M - r + h) \bmod M$. This event occurs with the probability $c_i(p) \frac{d_h(i, i_r)}{M}$.

where

$d_t(i, i_r) = (i_r - h - 1) \bmod M$ and $i_r = (M - r + h) \bmod M$

Therefore, the probability, $V_i^h(v)$, is given by

$$V_i^h(v) = (1 - C_i(p)) + c_i(p) \frac{d_h(i, i_r)}{M}$$

Now we proceed with the computation of the probability of resolution, α. Let $\sigma_{a_1,a_2}(h)$ be the probability that a conflict occurs in the resolution phase, given that the interval $(a_1,a_2]$ is searched and station h is the current virtual holder, where $a_k=(p_k-1)M+r_k$; k=1,2.

A conflict occurs, while the interval $(a_1,a_2]$ is searched, if more than one station contention parameter lie in the latter interval. Therefore, we can write

$$\sigma_{a_1,a_2}(h) = Pr[\text{ at least two } c_i\text{'s are in } (a_1,a_2], \text{ given that all } c_i\text{'s are in } (l,a_2]\ ]\ ,$$

where l is the lower bound of the initial PAI. This probability can be derived as:

$$\sigma_{a_1,a_2}(h)=\begin{cases}\prod_{i=0}^{M-1}L_i^h(a_2) - \sum_{i=0}^{M-1}\left((L_i^h(a_2)-L_i^h(a_1))\prod_{\substack{j=0\\ j\neq i}}^{M-1}L_j^h(a_1)\right) - \prod_{i=0}^{M-1}L_i^h(a_1) & \text{if } 0\le a_1 < a_2\\ 0 & \textit{otherwise}\end{cases}$$

where $L_i^h(x)=Pr[v_i\le x \mid H=h]=1-V_i^h(x)$.

Notice that the first term in the above expression represents the probability that all contention parameters c_i's are less than a_2, the second term represents the probability that exactly one c_i is in $(a_1,a_2]$ and the third term represents that all parameters c_i's are higher than a_1.

We can compute $\sigma(h)$, the probability that a collision occurs, given that the current virtual token holder is station h as follows:

$$\sigma(h)=\sum_{a_1\in S}\sum_{a_2=a_1+1}^{U}\sigma_{a_1,a_2}(h)\,Pr\,[\text{ the interval } (a_1,a_2] \text{ is searched }]$$

where $S=\{\,0\le n<U/n=(p-1)M+r;\ 0<p<m;\ 0\le r<M\,\}$ and $U=(m-1)M+M-1$.

Assuming the static approach, we can argue that the interval $(a_1, a_2]$ is searched if either a collision occurred in the interval $(2a_1-a_2, a_2]$, resulting in the update of the lower bound of the interval, or the channel remained idle when the interval $(a_1, 2a_2-a_1]$

was searched, resulting in the update of the upper bound of the interval. Therefore, we can write

$$\sigma(h) = \sum_{a_1 \in S} \sum_{a_2 = a_1 + 1}^{U} \sigma_{a_1,a_2}(h)\,(\sigma_{2a_1 - a_2, a_2}(h) + \delta_{a_1, 2a_2 - a_1}(h))$$

where

$\delta_{a,b}(h) = Pr$ [*Idle channel, given that* $(a,b]$ *is searched* and *h is the virtual token holder*]

However, the channel remains idle if there is no station i such that its contention parameter is in $(a,b]$, given that all contention parameters are less than b. Therefore, we can write

$$\delta_{a,b}(h) = \prod_{i=0}^{M-1} L_i^h(a)$$

Similarly, we can compute the probability, $\delta(h)$, that the channel remains idle as follows:

$$\delta(h) = \sum_{a_1 \in S} \sum_{a_2 = a_1 + 1}^{U} \delta_{a_1,a_2}(h)\,(\sigma_{2a_1 - a_2, a_2}(h) + \delta_{a_1, 2a_2 - a_1}(h))$$

We proceed to uncondition on H, to obtain

$$\sigma = \sum_{j=1}^{m-1} \sum_{h=0}^{M-1} \sigma(h) \frac{\lambda_{hj}}{\lambda}$$

$$\delta = \sum_{j=1}^{m} \sum_{h=0}^{M-1} \delta(h) \frac{\lambda_{hj}}{\lambda}$$

However, we know that a resolution is not achieved during the PAP, resulting in an additional step, if either the channel remains idle or a collision occurs. Therefore, the probability of resolution can be computed as $\alpha = 1 - (\sigma + \delta)$.

In the following section we compare the delay throughput characteristics of the proposed scheme with the results obtained by simulation.

6.5.3 Comparative Study

In order to evaluate the accuracy of the model in predicting the performance of the proposed protocol, simulation experiments were performed for three cases of network size: 5, 10 and 50 stations. In all cases, the channel bandwidth was 10,000,000 *bits/sec*, the message length was of constant length, 1000 bits, and the slot time was fixed at 0.02 *msec*. We assumed four priority classes of messages. The network traffic, a *Poisson* process, was equally distributed among stations and priority classes.

The results obtained by the simulation and the analytical model are shown in Figures 6.7, 6.8 and 6.9, for the three cases of network sizes.

We observe that the analytical model is able to predict the delay undergone by messages with reasonable accuracy. The results produced by the analysis are comparable to the results obtained by simulation. We also observe that in a heavy load situation and for a network with a small number of stations, the results of the analytical model tend to diverge from the results obtained by simulation. This can be attributed to the fact that when the load increases, the stations become ready all the time. Therefore, the effect of the reservation mechanism becomes significant and the stations will capture the channel in turn to transmit their messages. Thus, at heavy load the system approaches a token protocol. However, in our model we assumed the probability of a station to be the next virtual token holder, for a given network priority, depends only the station load. Consequently, as the load increases, the analytical model predicted a higher collision rate. This assumption introduces a slight margin of error which causes the divergence between the simulated results and the analytical results. Notice that the modeling of such a deterministic aspect of the protocol requires the inclusion of additional state variables in the system state vector. Although the derivation of the differential difference equations of the resulting Markovian process could be achieved without additional complexity, the solution of the resulting system becomes mathematically involved.

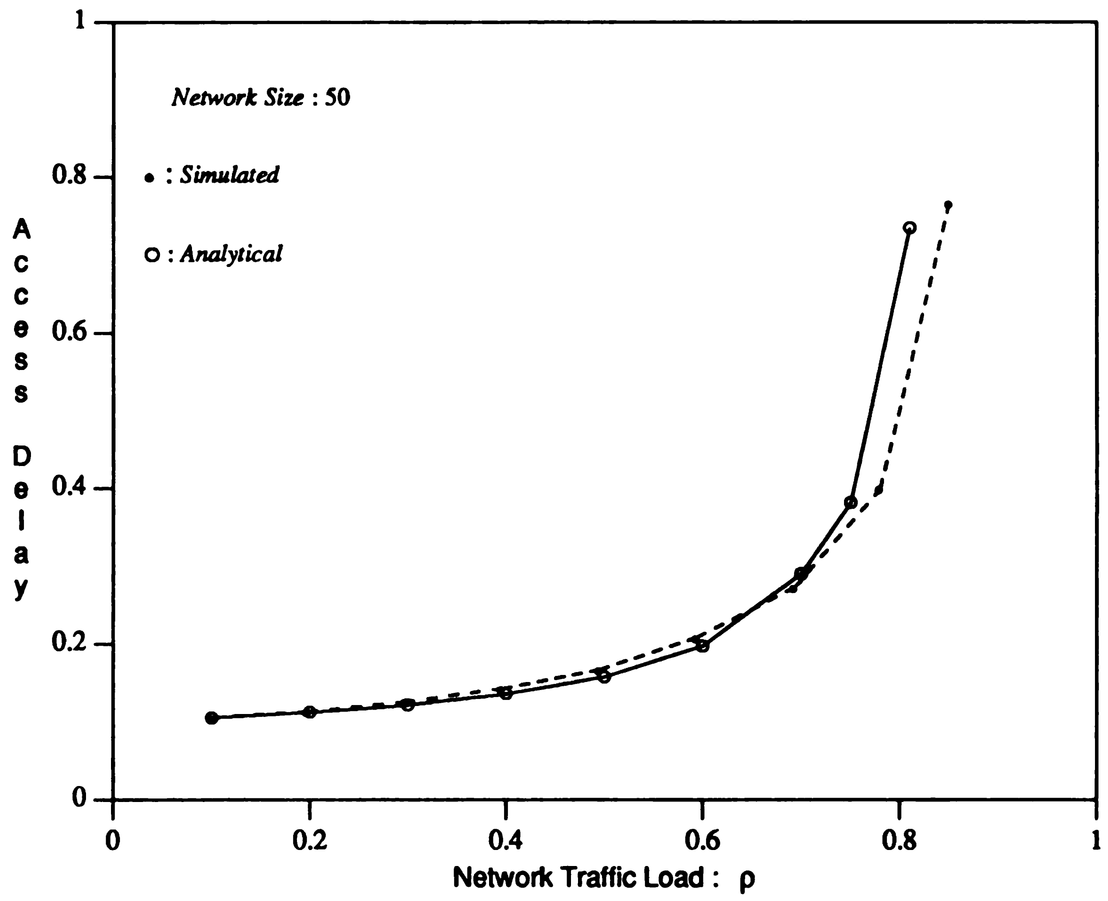

Figure 6.7 Network Delay Throughput Characteristics

6.6. CONCLUSION

Multiaccess protocols for bus networks are characterized by the interference problem inherent in the nature of the random access. We developed a model for the analysis of such protocols in a unified manner. The technique used for the queueing analysis is based on the supplementary variable approach. This method is shown to be tractable for the analysis of multiaccess protocols.

We established the set of equations which describe the behavior of the system and solved the model for the case where $\alpha(n)$ and $\beta(n)$ are constants. In the general solution of the model, the service requirements of the incoming message was drawn from a general distribution. Numerical results were computed for special cases of services. The mean

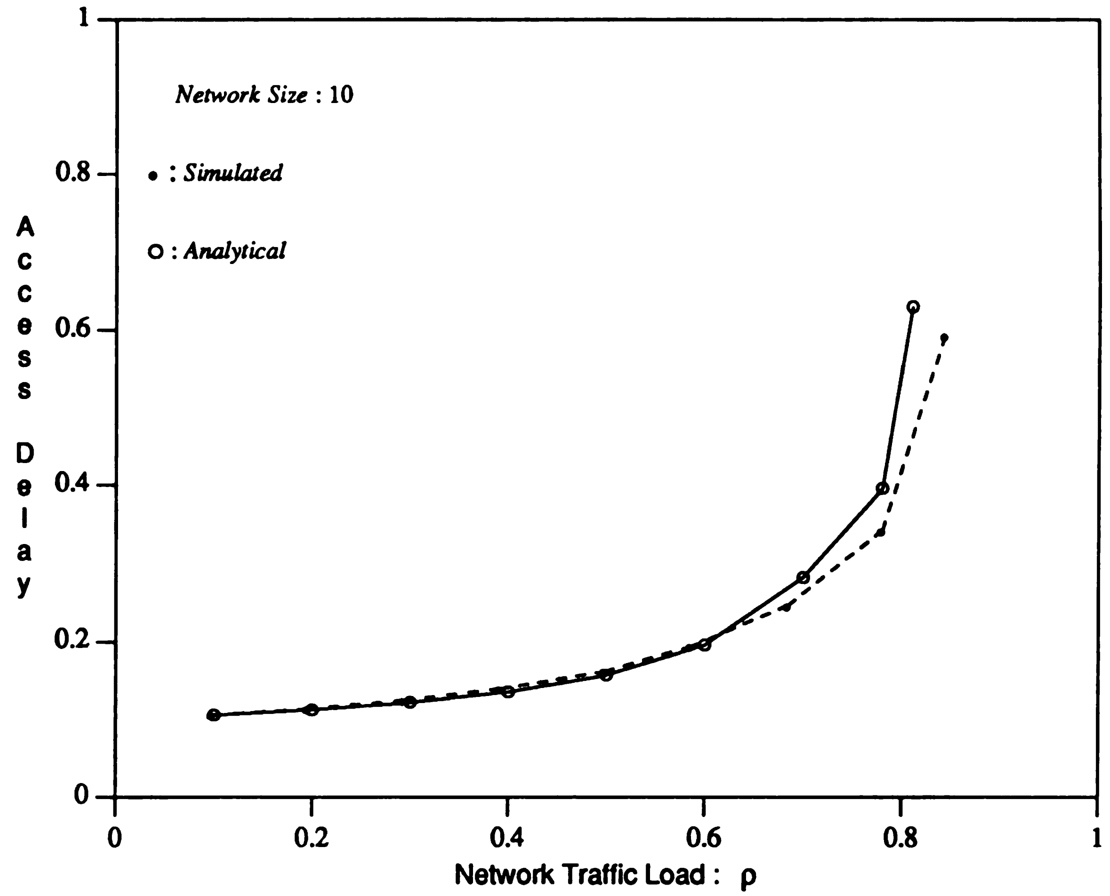

Figure 6.8 Network Delay Throughput Characteristics

queue length for different values of α and β, different network sizes and different service distribution was obtained. It was interresting to observe that the probability of collision has more significant effect on the mean queue length of contending stations than the probability of resolution. It was also observed that the service variability did not affect the performance of the system considerably. However, the results show that the system load has a strong impact on the performance of the protocol.

The application of the proposed model to the proposed protocol was investigated. We have shown that the analytical model could be used to predict the behavior of the protocol with a reasonable amount of accuracy.

Chapter 7

Conclusions and Directions for Future Research

This chapter describes my conclusions resulting from the research reported in this thesis. In addition, some implications of this work, specifically ways it might affect the design of priority schemes for local area networks, are given. Finally, directions for future research work are discussed.

The comments provided in this chapter are organized in three sections. In the first part, we summarize the main contributions of the research achieved in this dissertation. In the second part, we provide a brief summary of the research work described in this dissertation. In the third part, we discuss problems requiring further elaboration and describe some avenues for future research development. This should help the interested reader to note specific areas where further work is needed.

7.1 THESIS CONTRIBUTIONS

The main contributions of this research are the design of a protocol for local area networks that satisfies closely the requirements of a distributed real-time system and the development of an analytical model capable of comparing the performance of broadcasting-based protocols for bus networks in a unified manner.

An efficient prioritized multiaccess protocol for distributed real-time systems with bus interconnection topology has been proposed. The proposed protocol, based on the virtual token concept, is intended to remedy the shortcoming of the standard schemes, while retaining their main advantages. When the load is light, the proposed protocol behaves like the CSMA/CD protocol. When the load is heavy, however, the protocol behaves like the token-passing protocol. Furthermore, the proposed protocol reduces the unnecessary overhead generated by the token traveling between idle stations. Since the token is virtual, rather than physical, the occurrence of faulty situations due to the loss or duplication of the token is eliminated. The elimination of the physical token makes fault management far simpler than the standard token-passing scheme.

The simulation results have demonstrated that the proposed protocol performs closely to the ideal priority scheme. The overhead spent in implementing the protocol is relatively small. The results also indicate that the proposed protocol outperforms the standard schemes in a real-time environment.

The second part of the proposed research work was dedicated to developing a general analytical model to study the performance of protocols for bus networks. Our approach was based on the use of the supplementary variable technique. We have proven that this technique yields a tractable approach for analyzing the class of multiaccess protocols for bus networks. The main practical motivations come from providing an analytical tool to study and compare this class of protocols in a unified manner. For a given protocol, once its its probability of resolution, α, and its probability of conflict, β, are computed the analytical model could be used to determine its performance profile.

In the following section we provide a review of the contributions of the research work in a greater detail.

7.2 CONCLUSIONS

The main objective of Chapter 2 was to identify the design requirements of distributed real-time systems. An attempt to identify the fundamental problems involved in the design of such systems was made. We then considered the impact of the distributed real-system requirements on the design of local area network protocols intended to support the traffic generated by these systems. We finally discussed the suitability of the currently available schemes for real-time applications. The study showed that CSMA/CD is a simple multiaccess scheme and performs efficiently under light traffic conditions. It is also robust against transmission errors as it does not rely on critical control information. However, CSMA/CD based schemes cannot satisfactorily handle heavy traffic conditions. In addition, the protocol is not suitable for handling different classes of messages as it does not provide any priority functions.

The study also showed that the token passing bus protocol outperforms the CSMA/CD scheme in handling heavy traffic conditions. However, under light loads, token passing bus has longer mean delay time than CSMA/CD based protocols, since the token still has to visit idle stations before reaching a station ready to transmit. In addition, the priority scheme offered by the token passing protocol does not satisfy the evaluation criteria of a priority scheme. We concluded our discussion by observing that it is desirable to have a protocol which achieves good performance over the entire range of channel bandwidth and offers the capability to handle different classes of messages with different timing requirements. With this objective in mind, a novel priority-based multiaccess scheme was designed.

The description of the proposed protocol was provided in Chapter 3. The proposed protocol was an attempt to capture the essence of an ideal priority scheme with minimum overhead. Conflict-free access to the channel by stations at a given priority level is realized by establishing a link among all the stations currently holding a message of that priority, so that all these stations form a logical sub-ring. An implicit token travels from one ready station of the highest priority sub-ring to the next ready station of the same sub-ring, bypassing all idle stations. A contention based approach causes the virtual token to travel from a lower priority sub-ring to a higher priority sub-ring.

The description of the basic operations of this phase of the protocol and the methods that can be used to assess the current highest priority sub-ring were the objectives of Chapter 4. In this chapter, we proposed three approaches that could be used to implement the conflict resolution algorithm. The first method uses the load characteristics of the network stations to dynamically build an optimal binary tree. The second approach uses a simple dichotomy-based strategy to coordinate the search of the priority assessment interval. This method is static in the sense that it reproduces the same sequence of lower bounds for a given PAI. The third method is adaptive and uses the channel feedback to build a strategy based on the recent history of the channel.

The objective of Chapter 5 was to test the PAI searching strategies in some representative network situations and allow conclusion to be drawn about the effectiveness of each strategy. The results demonstrated that the static, dichotomy-based approach and the adaptive approach performed very closely. Furthermore, the results have shown that the dynamic approach produced better results than the other two methods; however, the improvement was not significant enough to justify the computational cost involved in the dynamic approach. Based on these results, the static approach was adopted for further analysis of the proposed protocol. A series of experiments were conducted to compare the performance of the proposed protocol to the performance of the standard schemes and to study the sensitivity of the protocol to certain network parameters. The results show that the proposed protocol fares better than the standard schemes. The results have also proven the capability of the protocol to handle different types of traffic with minimum overhead.

Chapter 6 presented the general analytical model which can be used to study the steady state behavior of the class of access mechanism in a bus networks. The proposed model is based on the supplementary variables approach. This approach was shown to result in a mathematically tractable set of state equations which describe the behavior of multiaccess protocols. Solutions for the resulting steady state equations were provided for both infinite and finite source models. We then applied the model to solve for the overall response time of the proposed scheme. We demonstrated that the results of the experimental model were not significantly different from the results of the analytical model.

7.3 FUTURE RESEARCH

The majority of real-time communications proposed in the literature associate priority functions with timing constraints. That is, deadline information is mapped into a priority level. These priorities are then used by the data link layer in regulating the access to the channel. However, very little work has addressed the problem of how these priorities are dynamically assigned to accurately reflect deadlines of currently active messages. It is clear that any dynamic message priority mapping scheme based strictly on deadlines is confronted with the challenging problem of the synchronization of the clocks at all stations of the network. Consequently, a study to identify other major design parameters and how they affect a distributed real-time application is required. A great deal of these parameters can be broadcast and made globally available for all stations without excessive overhead. The ability to understand the dynamic relationship between these variables and to control them optimally could be of great importance in the design of multiaccess schemes. The inclusion of the observed values of these parameters in the definition of the priority mapping function may improve the performance of the protocol and its robustness, in the sense that the protocol performance will not considerably deteriorate in the absence of perfect clock synchronization.

We proposed an analytical model to study the performance of multiaccess protocols. The model could be used to study and compare the performance profile of a large class of known multiaccess protocols such as random access protocols, adaptive CSMA/CD protocols and window protocols. A comparative study of these protocols could be achieved by evaluating the probability of collision and the probability of resolution of these schemes. Another possible extension of the model would be to include m supplementary variables, to keep complete information about the present history of the channel, in order to study the performance profile of each priority class. However, one becomes confronted with the problem of solving a set of differential difference equations in $m > 1$, a non-trivial problem even for small values of m. Simpler methods for studying differential difference equations resulting from non-Markovian processes with several states need to be investigated. The investigation of these methods will allow the development of the proper analytical tools to evaluate and accurately compare not only multiaccess protocols of the data link and physical layers, but also protocols at higher layers. These tools become more and more crucial as inter-process communication systems are proliferating rapidly.

BIBLIOGRAPHY

BIBLIOGRAPHY

[Abra70] Abramson, N., "The ALOHA system-Another alternative for computer communications", *In Proceedings of the AFIPS Fall joint Computer Conference*, AFIPS Press, Montvale, N.J.,, pp.281-285, 1970.

[Abra73] Abramson, N., "The Packet Switching With Satellites", *In Proceedings of the AFIPS National Computer Conference*, AFIPS Press, Montvale, N.J.,, pp.695-703, 1970.

[AhHU76] Aho, A.V., Hopcroft, J.E. and Ullman J.D., *The Design and Analysis of Computer Algorithms*, Addison Wesley Pub. Co., June 1976.

[Bux81] Bux, W., Closs, F., Janson, P.A., Kummerle, K., Muller, H.R., "A reliable token ring system for local area communication", *In Proceedings fo the national Telecommunications Conference*, New Orleans, La., Piscataway, N.J., pp. A2.2.1-A.2.2.6, 1981. *IEEE*,

[Cape79] "A Generalized TDMA: The multiple-accessing tree protocol" *IEEE Trans. Commun.*, COM-27, 1476-1484, Oct 1979.

[Carp84] Carpenter, R., "A comparison of two guaranteed local network access methods", *Data Communications*, Feb 1984.

[ChFr79] Chlamtac, I., Franta , W., "BRAM: The broadcast recognizing access method", *IEEE Trans. Commun*, COM-27, 1183-1190, Aug. 1979.

[Clar78] Clark, D. Progran, K. and Reed, D., "An introduction to local area networks", *Proc IEEE 66*, 1497-1516, Nov. 1978.

[Coxd54] Cox, D. R. "The analysis of Non-Markovian stochastic processes by the inclusion of supplementary variables", *Camb. Philos. Soc*, 51, 3, 433-441, Nov 1954.

[Dasa85] Dasarathy, B., "Timing Constraints of Real-Time Systems: Constructs for Expressing Them, Methods of Validating Them", *IEEE Trans. on Soft. Eng, SE-11,1*, pp.80-86, Jan. 1985.

[DiSM83] Dixon, R.C., Strole, N.C. and Markov, J.D., "A token-ring network for local data communications," IBM Systems Journal, Vol. 22, pp.47-62, January-February 1983.

[EsHS81] Eswaran, K.P., Hamacher, V.C., and Shedler, G.S., "Collision Free Access Control for Computer Communication Bus Network", IEEE Trans. Software Eng., Vol. SE-7, Nov. 1981.

[Efe78] Efe, K., "Heuristic Models of Task assignment scheduling in distributed systems", *IEEE computer*, Vol. 15, June 1982.

[FaNe69] Farmer, W.D., and Newhall, E.E., "An Experimental Distributed Switching System to Handle Bursty Computer Traffic", *In Proceedings of the ACM Symposium on Problems in the Optimization of Data Communications Systems* , Pine Mountain, Ga., Oct. 13-16., ACM, NEW YORK, PP. 1-33.

[FiTo84] Fine, M., and Tobagi, F.A., "Demand Assignment Multiple Access Schemes in Broadcast Bus Local Area Networks", *IEEE Transactions on Computers*, Vol. C-33, no. 12, Dec. 1984.

[FFHH73] Farber, D.J., Feldman, J., Heinrich, F., Hopwood, M., Larson, K., Loomis, D., and Rowe, L., 1973, "The Distributed Computing System ", *In Proceedings of the 7th Annual IEEE computer Society International Conference (COMPCON)*, San Fransisco, Calif., Feb.27-Mar.1, IEEE, Piscataway, N.J., pp. 31-34.

[FrBi80] Franta, W.R., and Bilodeau, H.B., "Analysis of a Prioritized CSMA Protocol Based on Stagered Delays", *Acta Informatica*, 13, Fasc 4, 299-324, 1980.

[FrBT81] Fratta, L., Borgonovo, F., Tobagi, F., "The EXPRESS_NET: A local area communication network integrating voice and data", *In Performance of data Communication Systems*, G. Pujolle, Ed. Elsevier, North-Holland, New York, pp. 77-88, 1981.

[Gall76] Gallager, R. G., "Basic limits on protocol information in data communication networks", *IEEE Trans Inf. Theory IT-22, 385-398, July 1976.*

[Gall78] Gallager, R. G., "Conflict resolution in random access broadcast networks", *In Proceedings of the AFOSR Workshop in Communication theory and Applications*, Provincetown, Mass., Sept. 17-20, 1978, *IEEE, Piscataway, N.J.*, pp. E3.6.1-E3.6.5.

[GrSH82] Grami, A., Sohraby, K., and Hayes, J., "Further results on probing", *In Proceedings of the International Communications Conference*, 1C.3.1-1C.3.3, Philadelphia, Pa., June 13-17, 1982.

[Hanse78] Hansen, P.B., "Distributed Processes: a Concurrent Programming Concept", *CACM*, Vol. 21, no. 11, Nov. 1978, pp. 934-941.

[Haye78] Hayes, J. F., "An adaptive technique for local distribution", *IEEE Trans. Commun*, COM-26, 1178-1186, Aug 1978.

[Hoar78] Hoar, C.A.R., "Communicating Sequential Processes", *CACM*, Vol. 21, no. 8, Aug. 1978, pp. 666-677.

[IBHK79] Ichbiah, J.D., Barnes, J.C., Heliard, J.C., Krieg-Brueckner, B., Roubine O., and Wichmann, B.A., "Rationale for the Design of the ADA Programming Language", *SIGPLAN Notices*, Vol. 14, no. 6, part B, Jun. 1979.

[IEEE82] IEEE Project 802 Local Area Network Standards, "Logical Link Control", *Draft D, IEEE Computer Society,* Nov. 1982.

[IEEE82] IEEE Project 802 Local Area Network Standards, "CSMA/CD Access Method and Physical Layer Specifications ", *Draft D, IEEE Computer Society,* Dec. 1982.

[IEEE83] IEEE Project 802 Local Area Network Standards, "Token-Passing Bus Access Methods and Physical Layer Specifications", *Draft IEEE Standard 802.4, Revision E,* July 1983.

[IIYO80] Iida, I., Ishizuka, M., Yasuda, Y. and Onoe, M., "Random access packet switched local computer network with priority function", *in Proc. of the IASTED int'l Sym., Applied Simulation and modeling (ASM'83),* May 1983.

[Insu85] Insup , L., "Language Constructs for Distributed Real-time Programming", *Proceedings on Real-time Systems Symposium*, 57-66, IEEE Computer Society Press, San Diego CA, 3-6 Dec. 1985.

[JaFi85] Jayasuman, A.P. and Fisher, P. D., "TSPS: A Token-Skipping Priority Scheme for Bus Networks",
Proc. of the 5th Int'l Conf, on Distributed Computing Systems, pp.56-63, 1985.

[Jens78] Jensen, E. D., "The Honeywell Expereimental Distributed Processor- An Overview of its Objectives, Philosophy and Architectural Facilities", *IEEE Computer*, Vol. 11, Jan 1978.

[JuWa84] Juang, J., Wah, B., "Unified Window Protocols for Contention resolution in Local Multi-access Networks", *Third Annual Joint Conference of the IEEE Computer and Communication Societies, San Fransisco CA, 1984.*

[Keil65] "The Role of Green's Function in Congestion Theory", *Proc. Symposium on Congestion Theory, Univ. of North Carolina*, Press, pp. 43-71, 1965.

[Kend53] "Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of Imbedded Markov Chain", *Annals of Mathematical of Statistics*, 24, pp. 338-354, 1953.

[Klei70] Kleinrock, L., "Analytic and Simulation Methods in Computer Network Design", *Proc. SJCC*, 569-579, 1970.

[Klei75] Kleinrock, L., "Queueing Systems", 1: Theory, New York: Jhon Wiley, 1975.

[Klei76] Kleinrock, L., "Queueing Systems", 2: Computer Applications, New York: Jhon Wiley, 1976.

[KlSc80] Kleinrock, L., and Scholl, M., "Packet switching in radio channels: New conflict-free multiple access schems", *IEEE Trans. Commun.*, COM-28, July 1980.

[KlTo75] Kleinrock, L., Tobagi, F. A., "Packet Switching in radio channels: Part I-Carrier sense multiple access and their throughput-delay characteristics", *IEEE Trans. Commu.*, Comm-23, Dec 12, 1975. pp.1400-1416.

[KlYe78] Kleinrock, L., and Yemini, Y., "An Optimal Adaptive Scheme for Multiple Access Broadcast Communication", *Proc. ICC*, 7.2.1-7.2.5, 1978.

[Knut73] "The Art of Computer Programming", *Vol. 3, Sorting and Searching*, Addison Wesley, 1973.

[Kuro84] Kurose J. F., "Time-constrained communication in multiple access networks", *Ph.D. Dissertation*, Department of Computer Science, Columbia University, July 1984.

[KuSY84] Kurose , J.F., Schwartz , M., and Yemini, Y., "Multiple-Access Protocols and Time-Constrained Communication", *Computing Surveys*, 16 ,1, 43-70, March 1984.

[Lam83] Lam, S. , Q*Computer Communication Networks", *Vol I, Principles*, Chap 4, pp.115-155, Editor Wushow, C., Prentice Hall, 1983.

[LaMp85] Lamport, L., Melliar-Smith, P.M., "Synchronization Clocks in the Presence of Faults", *Journal ACM,* Vol. 32, 1, January 1985, pp. 52-78.

[Lee84] Lee, I., ''A Programming System for Distributed Real-Time Applications'', *Tech. Report MS-CIS-84-51*, University of Penn, Sept. 1984.

[LeLa83] LeLann, G. ''On Real-Time Distributed Computing'', *Invited paper IFIP Congress*, pp.741-753, North Holland, Sept 1983.

[LeLa85] LeLann, G., ''Issues In Real-Time Local Area Networks'', *10th Conference on Local Computer Networks*, 1-18, Minnesota, Oct 1985.

[LeLa87] ''The 802.3D Protocol: A Variation of the IEEE802.3 Standard For Real-Time Lan's'', *An Introduction*, July 1987, INRIA.

[LGKN86] Liu, Y.H., Gendreau, T.B., King, C.T. and Ni, L.M., ''A Session Layer Design of a Reliable IPC System in the UNIX 4.2 Environment", *Proc. of the 1986 Computer Networking Symposium*, pp.120-129, 1986.

[LiFl82] Limb , J,. Flores, C., ''Description of FASNET- A Unidirectional Local-Area Communication Network'', *Bell Syst. Tech. J. 61, 7*, pp. 1413-1440. 1982.

[LiHG82] Li, L., Hughes, H., and Greenberg, L., ''Performance Analysis of a Shortest Delay Protocol'', *Computer Networks*, 6, 1 , July 1982.

[LiLa73] Liu, C.L., and Layland, J.W., ''Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment'', *Journal ACM*, 20, 46-61, Jan 1973.

[Lind52] Lindley, D.V., ''The Theory of Queues with a Single Server'', *Proc. Cambridge Philosophical Society*, 48, pp. 277-289, 1952.

[LiRo84] Liu, M.T. and Rouse, D.M., ''A Study of Ring Networks'', *Ring Technology Local Area Networks*, edited by I. N. Dallas and E. B. Spratt, North-Holland Pub. Co., 1984, pp.1-39.

[Liu85] Liu, M.T., ''Introduction - Distributed Computing'', *IEEE Trans. on Computers*, December 1985, pp.1069-1071.

[MeBO76] Metacalfe, R. M., Boggs, D. R., Q*Ethernet: Distributed Packet Switching for Local Computer Networks'', *Comm. ACM 19*, July 7, 1976, pp.395-403.

[Meye80] Meyer, J.F., "On Evaluating the Performability of Degradable Computing Systems", *UCLA Report no. CSD-810730,* July 1981.

[Moka83] Mok, A.K., "Fundamental Design Problems of Distributed System for the hard real-time environment", *Ph.D. Th.*, MIT, MIT/LCS/TR-297, May 1983.

[Moka85] Mok, A.K., "The Design of Real-Time Programming Systems Based on Process Models", *Proc. Real-Time Systems Symposium*, pp.5-17, Dec 1985.

[NiGe86] Ni, L. M., and Gendreau T. B., "A Universal Interprocess Communication System for Distributed Computing", *Technical Report, Dept. of Computer Science, Michigan State University*, 1986.

[NiLi83] Ni, M. L., and Li, X., "A Prioritized Packet Transmission in Local Multiaccess Networks" *Proc. of the 8th Data Communications Symposium*, October 1983, pp.234-244.

[Parn75] Parnas, D. L. , " On the criteria to be used decomposing systems into modules " *Proc. ACM Internal Conf. Reliable Software, 1975, ACM, New York*, 1975,pp.294-304

[PoMi83] Powell, M.L., and Miller, B.P., "Process Migration in Demos/MP", *Proc. of the Ninth Symp. on Operating System Principles (ACM)*, 110-119, 1983.

[RaJa69] Rao, S., and Jaiswal N., "On a class of queueing problems and discrete transforms", *Journal of royal statistics*, 1062-1076, 1969.

[RaLC78] Randell, B., Lee , P.A., and C., P.C., "Reliability issues in computing system design", *ACM Computing Surveys*, 10, 123-165, June 1978.

[RoTo81] Rom, R., and Tobagi, F.A., "Message-based priority functions in local multi-access communication systems", *Computer Networks*, 273-286, North Holland, 1981.

[Schw77] Schwartz, M. "Computer Communication Network Design and Analysis", *Prentice- Hall*, New York, 1977.

[Schw85] Schwetman, H.D., "CSIM: A C-Based, Process-Oriented Simulation Language", *Microelectronics and Computer Technology Corporation*, Technical Report, PP-080-85.

[SDRC82] Shoch, J. F., Dalal, Y.K. , Redell, D.D.,and Crane, R.C., "Evolution fo the Ethernet Local Computer Network", *Computer*, Vol. 15, no. 8, pp. 10-27, 1982.

[Span79] Spaniol, O., "Modeling of Local Area Networks", *Computer Networks*, Vol. 3, pp. 315-326, 1979.

[Stac80] Stack , T.R., "Protocols for Local Area Networks", *Proc IEEE Trends and Applications : Computer Network Protocols*, 83-93, May 1980.

[Stal84] Stallings, W., *Local Networks: An Introduction*, Macmillan Pub. Co., 1985.

[Stan84] Stankovich, J.A., "A Perspective on Distributed Computer Systems", *IEEE Trans. on Computers*, Vol. c-33, no. 12, Dec. 1984.

[Tack65] "Combinatorial Methods in the Theory of Stochastic Processes", Wiley (New York), 1967.

[TaMB80] Taylor, D. J., Morgan D. E., Black, J. P., "Redundancy in data structure: Improving software fault tolerance" *IEEE Trans. on Soft. Engineering, Vol. SE-6, NO. 6*, November 1980.

[Toba82] Tobagi, F. A., "Carrier-sense multiple access with message-based priority functions", *IEEE Transactions on Communications*, Com-30, 185-200, Jan 1982.

[ToHu80] Tobagi, F.A., and Hunt, V.B., "Performance analysis of carrier sense multiple access with collision detection", *Comput. Networks*, 4, Oct/Nov. 1980.

[ToRo80] Tobagi, F.A., and Rom, R., "Efficient round-robin and priority schemes for Unidirectional broadcast systems", *Local Networks for Computer Communications*, 125-138, North-Holland/IFIP, 1980.

[ToVe82] Towsley , D., Venkatesh, G., "Window random access protocol for local computer networks", *IEEE Trans. Comput. C-31*, Aug. 1982, pp.715-722.

[WaJu83] Wah, B., and Juang, J., "An Efficient Protocol for Load Balancing on CSMA/CD Networks", *Proceedings of 8th Conference on Local Computer Networks, Oct. 1983.*

[WiTi80] Witte, L.D., and Tilborg, V., "MICROS, A Distributed Operating System for MICRONET, A Reconfigurable Network Computer", *IEEE Trans. on Computers*, 1133-1144, Dec. 1980.

[ZhSR87] Zhao, W., Stankovic, J.A., and Ramamritham, K., "A Window Protocol for Transmission of Time Constrained Messages", *COINS Tech. Report 87-110*, Nov 87.

[Zimm80] Zimmermann, H., "OSI Reference Model-The ISO Model of Architecture for Open Systems interconnection", *IEEE Trans. Commun.*, COM-28, 425-432, April 1980.

[ZnBa81] Znati, T. and Baccelli, F., "Queueing Systems With Breakdowns And Data Base Modeling", Performance '81, F.J. Kylstra, North-Holland Publishing Company, pp. 213,231.

[ZnNi87] Znati, T. and Ni, L., "A Prioritized Multiaccess Protocol For Real-Time Applications", *Proceedings of the 7th International Conference on Distributed Computing Systems*, Berlin, West Germany, Sep. 21-25 1987, pp. 324-331.

[ZnNi88] Znati, T. and Ni, L., "A Performance Modeling of Distributed Multiaccess Protocols", *In Proceedings on 7th Annual Joint Conference of the IEEE Computer and Communications Societies, Networks: Evolution or Revolution*, IEEE INFOCOM'88, New Orleans, March 27-31 1988, pp. 9B4.1-9B4.10,